

## ГЛАВА XX. Кластеризация

*Меньшинство – это совокупность лиц,  
выделенных особыми качествами;  
масса – не выделенных ничем.*

Х. Ортега-и-Гассет

*Религия объединяет народ,  
но разъединяет народы.*

Л.М. Рошаль

**Кластеризация** – это разбиение множества объектов на группы похожих, которые и называют **кластерами**. Такая постановка задачи относится к обучению без меток (Unsupervised Learning): есть только множество объектов и надо понять, как оно устроено; в данном случае – как разбивается на кластеры. Данное выше определение очень неформальное, иногда его дополняют пояснениями, что объекты внутри кластера должны располагаться недалеко друг от друга (требование **маленьких внутрикластерных расстояний**), а сами кластеры – быть удалёнными друг от друга (требование **больших межкластерных расстояний**). Как мы увидим дальше, это пояснение не охватывает все практически важные случаи кластеризации, опирается на расстояния между объектами и между кластерами (последнее надо формализовать), а также оперирует понятиями «маленькие / большие», что также нуждается в формализации. Алгоритмы для кластеризации называют **кластеризаторами**.

В задачах кластеризации можно предполагать, что объекты заданы признаковыми описаниями (**кластеризация на признаках – feature-based clustering**), тогда часто на них придётся ввести метрику<sup>1</sup>, или с помощью попарных сходств/различий (**кластеризация на сходствах/различиях – dis/similarity-based clustering**).

На рис. XX.1 показан пример множества объектов для кластеризации, а на рис. XX.2 возможные варианты решения. Заметим, что **мы ничего не сказали про число кластеров**: на сколько групп надо разбивать объекты. Как мы

---

<sup>1</sup> Адекватную метрику, при которой решение задачи имеет смысл.

увидим дальше, некоторые методы автоматически определяют число кластеров, для некоторых это число является гиперпараметром. Сложно сказать, какое из представленных разбиений (а они как раз отличаются числом кластеров) логично. Более того, даже при фиксированном числе кластеров и евклидовом расстоянии между объектами не всегда понятно, как разбивать объекты, т.е. задача неинтуитивна для человека и может не иметь правильного решения (в отличие от задач с размеченными данными – там есть истинные целевые значения).

В кластеризации может быть много допустимых решений, часто нет правильного!

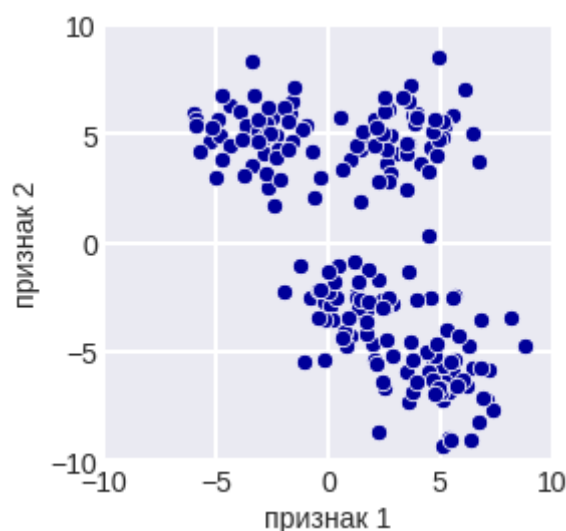


Рис. XX.1. Пример множества объектов (точечной конфигурации) для задачи кластеризации.

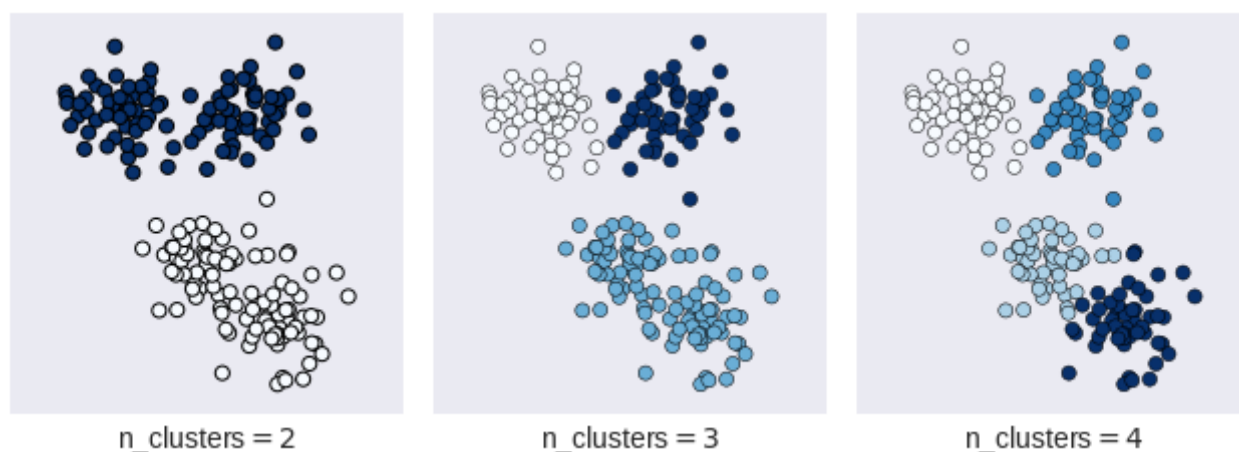


Рис. XX.2. Варианты решения задачи кластеризации при разном числе кластеров.

Кластеризация бывает **плоской** / **разделяющей** (**flat** / **partitional**), когда деление происходит на фиксированное число  $k$  кластеров (между ними не

предполагается какой-то связи), при этом выделяют **чёткую (hard)** кластеризацию, если кластеры не пересекаются:

$$\mathcal{X} = C_1 \cup \dots \cup C_k, i \neq j \Rightarrow C_i \cap C_j = \emptyset,$$

здесь  $\mathcal{X}$  – пространство объектов, и **нечёткую / мягкую (fuzzy / soft)** кластеризацию, если каждый  $i$ -й объект  $x_i$  может принадлежать каждому  $j$ -му кластеру с некоторой степенью принадлежности  $r_{it} \in [0, 1]$ , могут быть какие-то ограничения на степени принадлежности, например нормирующее ограничение:

$$\forall i \in \{1, 2, \dots, m\} \sum_{t=1}^k r_{it} = 1.$$

В этом случае алгоритм кластеризации выдаёт  $m \times k$ -матрицу степеней принадлежности  $\|r_{it}\|$ . Также бывает **иерархическая (hierarchical) кластеризация**, когда все данные образуют один кластер и далее рекурсивно этот кластер распадается на подкластеры (представляется в виде объединения непересекающихся кластеров), каждый из которых также может распадаться и т.д. Получается что-то типа системы каталогов файлов в операционных системах, только теперь в таких каталогах находятся наши объекты., см. рис. XX.3. Отметим, что организация данных и знаний в таком виде традиционна для различных областей интересов человека, например классификация живых организмов тоже иерархическая<sup>1</sup>.

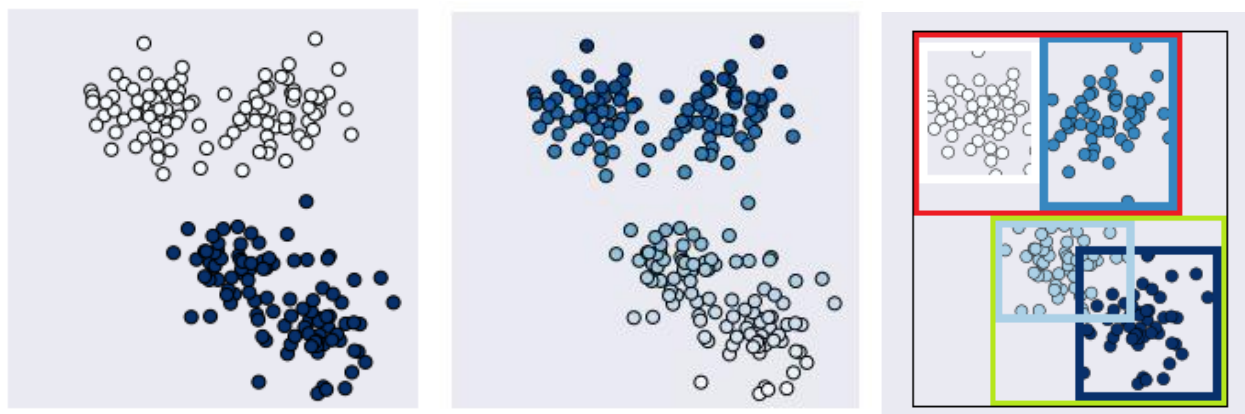


Рис. XX.3. Примеры кластеризаций, слева-направо: чёткая плоская (разным цветом показаны разные кластеры), нечёткая (интенсивностью цвета показана принадлежность к одному из двух кластеров), иерархическая (прямоугольниками показаны «каталоги»).

<sup>1</sup> [https://ru.wikipedia.org/wiki/Биологическая\\_систематика](https://ru.wikipedia.org/wiki/Биологическая_систематика)

Приведём некоторые примеры – **зачем на практике делать кластеризацию**. Отметим, что она редко бывает самостоятельной задачей, от которой есть какая-то бизнес-польза, но

- является важной составляющей решений других задач (semi-supervised, outlier detection, active learning), встречается в семплировании и т.п.,
- применяется для кластеризации клиентов / товаров (например, для «затачивания» алгоритмов под определённую группу клиентов или прогнозирования спроса на отдельные группы товаров). Все клиенты и товары могут быть слишком большой и разнородной группой, а отдельные кластеры – содержать достаточно объектов для обучения алгоритмов и при этом не быть слишком разнородными,
- применяется для создания иерархии сущностей: организации системы каталогов данных для того, чтобы попытаться разобраться, как они устроены<sup>1</sup>.
- полезна для сжатия данных – чтобы хранить не все объекты, а только представителей кластеров; для экономии времени запускать алгоритм не на всех объектах кластера, а на нескольких его представителях,
- необходима для нахождения сообществ в социальных сетях (community detection), в принципе это отдельная задача, которая важна для эффективного распространения новостей, рекламы, предложения услуг.
- помогает в анализе данных, например для нахождения групп похожих признаков (только здесь объектами являются признаки).

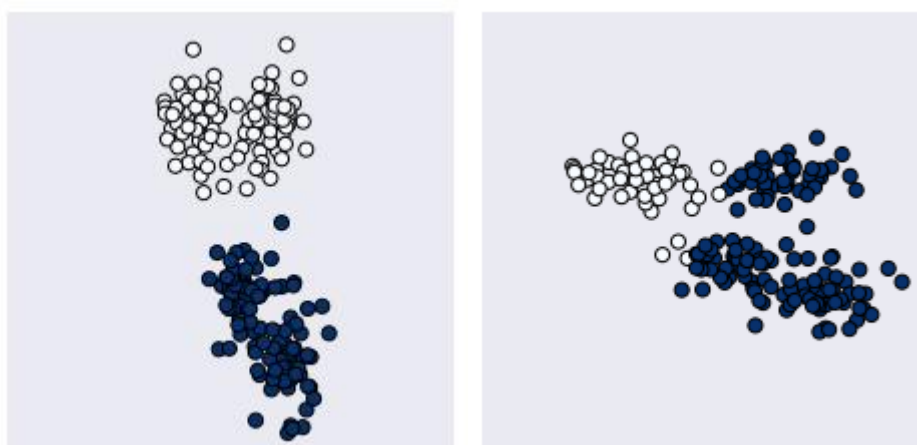


Рис. XX.4. Пример кластеризации при разных масштабах признаков.

<sup>1</sup> Вообще, классическая идея математики – факторизация – как раз заключается в переходе к более крупным сущностям.

Дальше в описании алгоритмов будем использовать метрику, считаем, что она выбрана очень грамотно и отражает семантическое различие объектов. На рис. XX.4 показан результат одного и того же алгоритма кластеризации, который использует евклидово расстояние в признаковом пространстве, при этом используется разный масштаб признаков (видно, что на левом рисунке точки вытянуты по вертикали, а на правом – по горизонтали). В результате при использовании одного и того же кластеризатора получаются разные кластеризации. При кластеризации часто используют предобработку данных типа стандартизации<sup>1</sup>.

Как всегда, если используется метрика, значит она должна отражать семантическое различие объектов!

Зададимся вопросом – как сделать кластеризацию? Малость внутрикластерных расстояний можно формализовать задачей типа

$$\sum_t v_t \sum_{x_i, x_j \in C_t} \rho(x_i, x_j)^\gamma \rightarrow \min_{\{C_t\}}.$$

В коэффициент  $v_t$  может быть заложена нормировка:

$$v_t = \frac{1}{|C_t|},$$

$\gamma$  – также гиперпараметр метода, который при увеличении больше штрафует за большие расстояния между объектами в одном кластере,  $\{C_t\}$  – искомое разбиение на кластеры. Во многих частных случаях такая задача NP-полная. Попробуем её упростить (точнее даже заметь другой), пусть с каждым кластером  $C_t$  ассоциирована некоторая точка  $\mu_t$  (можно назвать её центром, далее будет понятно почему), решим такую задачу оптимизации

$$\text{inertia} = \sum_t \sum_{x_i \in C_t} \|x_i - \mu_t\|^2 \rightarrow \min_{\{C_t\}, \{\mu_t\}}. \quad (\text{XX.1})$$

Здесь оптимизация происходит по разбиениям на кластеры  $\{C_t\}$  и по выбору положений центров  $\{\mu_t\}$ . Если зафиксировать кластеризацию  $\{C_t\}$ , то оптимальное решение по выбору параметров  $\{\mu_t\}$  – каждая точка  $\mu_t$  совпадает с центроидом кластера:

<sup>1</sup> Как всегда, она ничего не гарантирует. Лучше подбирать масштаб признаков. Также метрики типа евклидовой лучше применять в однородных признаковых пространствах.

$$\mu_t = \frac{1}{|C_t|} \sum_{x_i \in C_t} x_i$$

(именно поэтому мы и называли её центром). Если зафиксировать центры  $\{\mu_t\}$ , то оптимальное решение по выбору кластеризации  $\{C_t\}$  – кластер формируют ближайшие к центру объекты<sup>1</sup>:

$$C_t = \{i \mid \|x_i - \mu_t\| = \min_j \|x_i - \mu_j\|\}.$$

Можно сделать итеративную минимизацию: фиксировать одну группу параметров, тогда вторая группа определяется автоматически. Приходим к описанию алгоритма ***k*-средних (*k*-means, Lloyd's algorithm)**<sup>2</sup>.

Довольно частый приём: фиксируется одна группа параметров и оптимизация производится по другой.

Вход алгоритма: выборка  $\{x_1, \dots, x_m\} \subseteq \mathbb{R}^n$ . Гиперпараметр алгоритма: натуральное число  $k$  (число кластеров для построения).

- Инициализация: выбираем случайно  $k$  центров кластеров:

$$\{\mu_1, \dots, \mu_k\} \subseteq \mathbb{R}^n.$$

- Итерация алгоритма (итерации повторяются, пока центры кластеров не стабилизируются):

- Этап приписывания (assignment): каждый объект приписать к тому кластеру, к центру которого он ближе:

$$C_t = \{i \mid \|x_i - \mu_t\| = \min_j \|x_i - \mu_j\|\}.$$

- Этап пересчёта (update): пересчитать центры кластеров:

$$\mu_t = \frac{1}{|C_t|} \sum_{x_i \in C_t} x_i.$$

<sup>1</sup> Для удобства, множества  $C_t$  мы составляем из номеров объектов, при этом говорим, что «объекты формируют кластер».

<sup>2</sup> Есть несколько вариантов описания *k*-средних, здесь представлен т.н. алгоритм Ллойда. Lloyd S. Least squares quantization in PCM //IEEE transactions on information theory. – 1982. – Т. 28. – №. 2. – С. 129-137. Термин «*k*-means» введён в MacQueen J. et al. Some methods for classification and analysis of multivariate observations //Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. – 1967. – Т. 1. – №. 14. – С. 281-297.

На этапе инициализации **в качестве центров лучше брать случайные объекты**. На этапе приписывания формально может быть «ничья»: когда объект одинаково близок к нескольким центрам, тогда его случайно относят в один из кластеров, соответствующих этим центрам. Формально на этапе пересчёта может быть деление на ноль, если какой-то кластер оказался пустым:  $C_t = \emptyset$  (для страховки от этого случая мы выбирали в качестве центров объекты), тогда положение соответствующего центра не меняется или переназначается случайно.

На рис. XX.5 показано, как по итерациям меняются положения центров (крупные квадраты) и состав кластеров (объекты одного кластера раскрашены одним цветом) при обучении метода  $k$ -средних. На следующих итерациях центры и кластеры не меняются – наступает стабилизация. Очевидно, что результат работы алгоритма зависит от начальной инициализации, см. рис. XX.6. В современных реализациях алгоритм запускается несколько ( $n_{\text{init}}^1$ ) раз и выбирается лучшее решение по некоторому встроенному показателю качества, например по значению (XX.1), которое теперь можно переписать так:

$$\text{inertia} = \sum_{i=1}^m \min_t \|x_i - \mu_t\|^2.$$

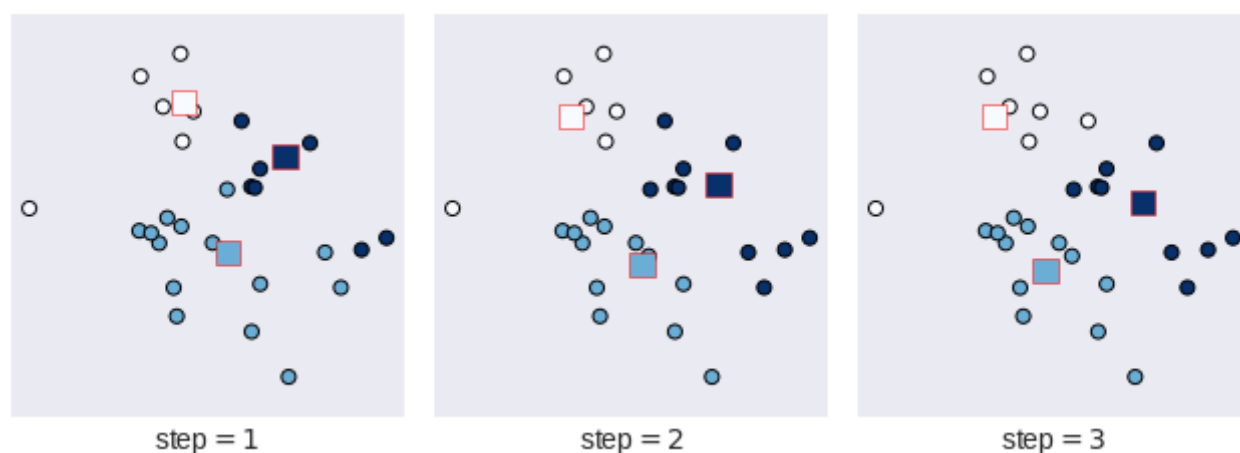


Рис. XX.5. Итерации метода  $k$ -средних.

<sup>1</sup> Здесь и далее указаны названия гиперпараметров в реализации библиотеки `sklearn`.



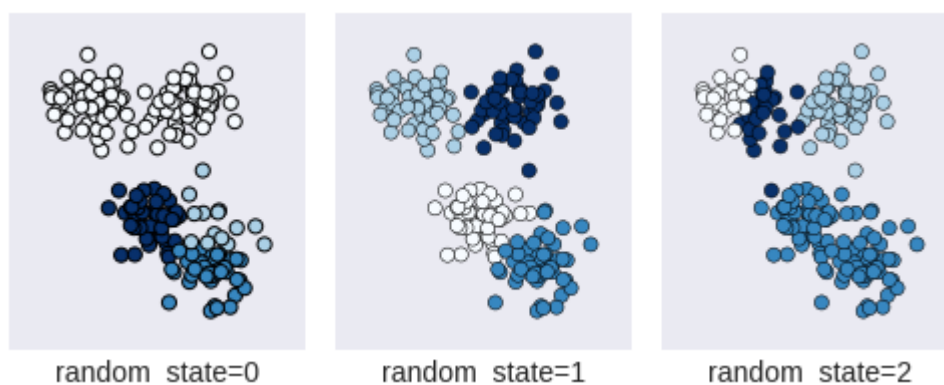


Рис. XX.6. Результат работы  $k$ -means на одном и том же датасете при разных начальных инициализациях.

Алгоритм обучается пока параметры не станут меняться на величины меньше некоторого порога (tol) или число итераций не превысит ограничение (max\_iter). Есть также версия алгоритма  **$k$ -Means с мини-батчами**<sup>1</sup>, в которой положение центра  $\mu_i$  корректируется в сторону объекта  $x_i$  из батча (подмножества выборки), который [объект] к нему ближе:

$$\mu_i \leftarrow \mu_i + \eta(x_i - \mu_i).$$

Эта версия предназначена для кластеризации больших наборов данных, в ней часто гиперпараметр n\_init определяет не число запусков, а число случайных начальных инициализаций, из которых выбирается лучшая для единственного запуска основного алгоритма (например, по значению (XX.1)). На рис. XX.7 продемонстрировано, что результаты алгоритмов  $k$ -средних в классической и в мини-батч-версии похожи, но есть небольшие различия на границах кластеров.



Рис. XX.7. Результат кластеризации  $k$ -means (слева),  $k$ -means по батчам (по центру), объекты, на которых кластеризация не совпала (выделены справа<sup>2</sup>).

<sup>1</sup> D. Sculley «Web-Scale K-Means Clustering» // WWW 2010: Proceedings of the 19th Annual International World Wide Web Conference. April, 2010. <http://www.eecs.tufts.edu/~dsculley/papers/fastkmeans.pdf>

<sup>2</sup> [https://scikit-learn.org/stable/auto\\_examples/cluster/plot\\_mini\\_batch\\_kmeans.html](https://scikit-learn.org/stable/auto_examples/cluster/plot_mini_batch_kmeans.html)



В алгоритме  $k$ -means число кластеров  $k$  ( $n\_clusters^1$ ) является гиперпараметром и задаётся до обучения. На рис. XX.8 показано, как работает кластеризатор при различных значениях  $k$ . Универсальной стратегии выбора значения  $k$  не существуют. Нетрудно понять, что при увеличении числа кластеров уменьшается показатель  $inertia$ , он гарантированно обращается в ноль, когда число кластеров равно числу объектов, см. рис. XX.9. Но график часто напоминает «согнутую руку», в **методе локтя**<sup>2</sup> выбирается значение  $k$ , которое соответствует примерному положению локтя этой руки. Например, на рис. XX.9 это  $k=3$ , что соответствует визуальному восприятию (график построен для системы точек с рис. XX.2).

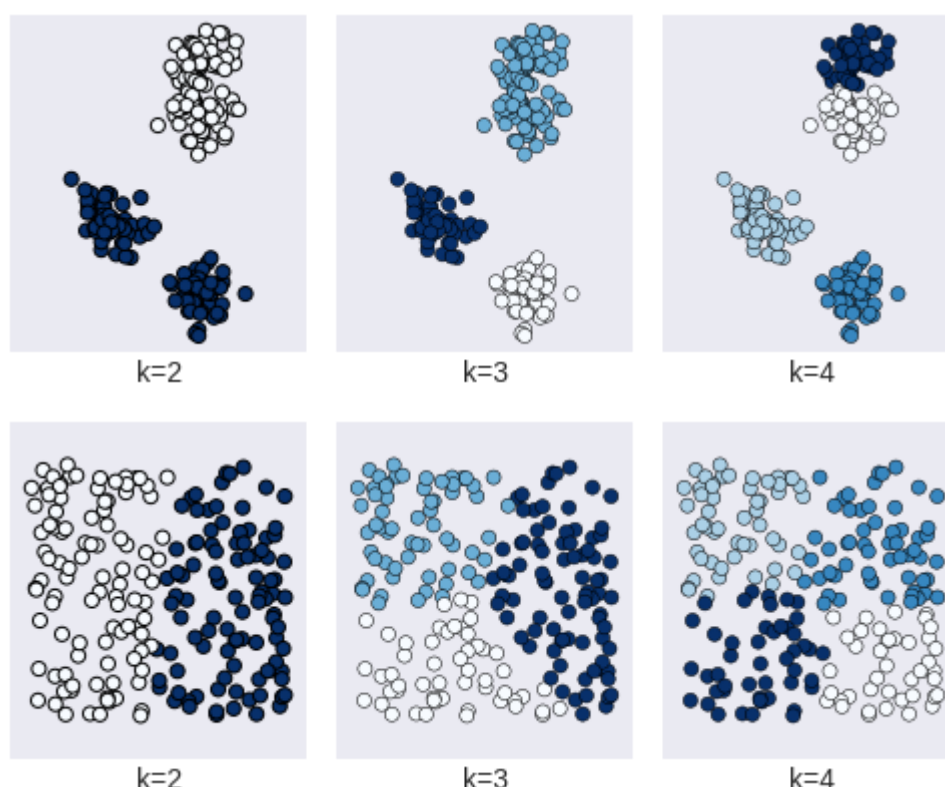


Рис. XX.8. Результаты алгоритма  $k$ -средних при разном числе кластеров  $k$ .

Теоретически метод  $k$ -средних может сходиться долго, средняя сложность —  $O(kmn_{iter})$ , где  $n_{iter}$  — число итераций, в худшем случае —  $O(m^{(k+2/n)})$ , но на практике он достаточно быстрый<sup>3</sup>.

<sup>1</sup> Интересно, что в реализации sklearn алгоритм называется KMeans, а основной гиперпараметр —  $n\_clusters$ .

<sup>2</sup> Thorndike R. L. Who belongs in the family? //Psychometrika. — 1953. — Т. 18. — №. 4. — С. 267-276.

<sup>3</sup> Arthur D., Vassilvitskii S. How slow is the k-means method? //Proceedings of the twenty-second annual symposium on Computational geometry. — 2006. — С. 144-153.

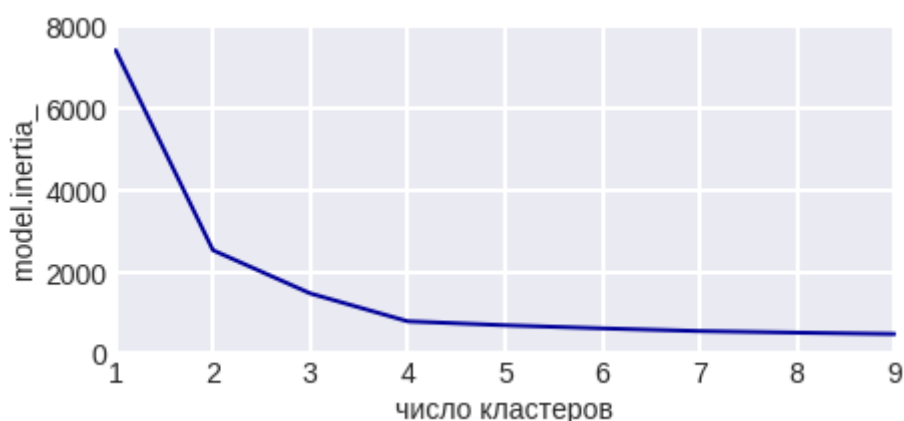


Рис. XX.9. Качество кластеризации при различном числе кластеров.

Для выбора начальных центров есть различные эвристики. Как уже упоминали, логично выбирать центрами какие-то объекты, тогда на первой итерации не будет пустых кластеров, также логично выбирать центры подальше друг от друга, что и делает **эвристика k-means++**: 1й центр – один из объектов, 2й – максимально удалённый от первого, 3й – максимально удалённый от предыдущих центров:

$$\mu_3 = \arg \max_{x_i} \min[\|x_i - \mu_1\|^2, \|x_i - \mu_2\|^2]$$

и т.д. Чаще применяют вероятностный выбор, чтобы при перезапусках могли быть разные ответы: вероятность выбрать точку  $x_i$  в качестве очередного  $(t+1)$ -го центра пропорциональна значению

$$\min\{\|x_i - \mu_j\|^2\}_{j=1}^t.$$

Алгоритм  $k$ -средних вообще допускает много модификаций, например, можно оценивать центры более робастным способом. Можно использовать ядерные деформации (kernel tricks) в  $k$ -means (для кластеризации в более сложном пространстве).

Классический **алгоритм  $k$ -средних хорошо работает для примерно равномогных кластеров «шаровой формы»** (позже мы лучше поймём почему). Оптимизируемый функционал (XX.1) у него невыпуклый, рис. XX.10 иллюстрирует оба данных утверждения: оптимизационный алгоритм сошёлся к двум разным решениям, которые не соответствуют нашей интуиции, как надо кластеризовать данную конфигурацию точек. На подобных точечных конфигурациях из «ленточных» кластеров кластеризатор  $k$ -means работает плохо.

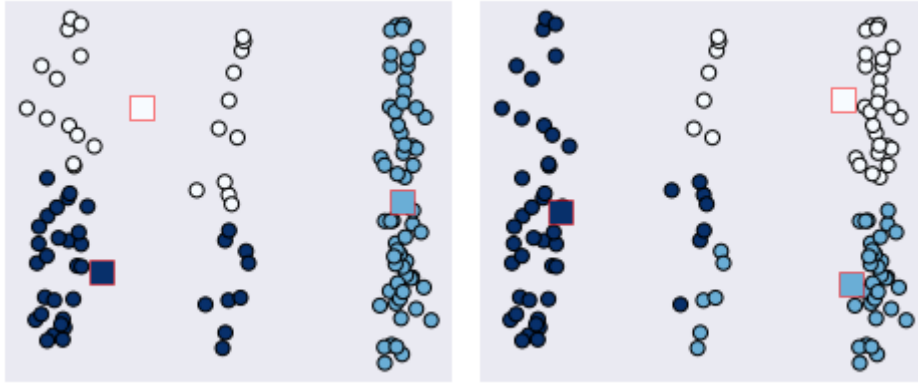


Рис. XX.10. Пример результата алгоритма  $k$ -средних для конфигурации «три ленты».

### Мягкий алгоритм $k$ -средних (soft $k$ -means)

Для обобщения алгоритма  $k$ -means на случай нечёткой кластеризации немного перепишем некоторые формулы. Вместо формулы

$$C_t = \{i \mid \|x_i - \mu_t\| = \min_j \|x_i - \mu_j\|\}$$

для обозначения приписывания объектов к кластеру с ближайшим центром  $\mu_t$ , введём индикатор принадлежности объекта  $x_i$  к кластеру  $C_t$

$$r_{it} = I[\|x_i - \mu_t\| = \min_s \|x_i - \mu_s\|]$$

(считаем, что минимум единственный). Таким образом

$$r_{it} = 1 \Leftrightarrow x_i \in C_t,$$

тогда пересчёт центров кластеров

$$\mu_t = \frac{1}{|C_t|} \sum_{i \in C_t} x_i$$

перепишется в виде

$$\mu_t = \frac{1}{\sum_{i=1}^m r_{it}} \sum_{i=1}^m r_{it} x_i.$$

Частный приём: переписываем формулу, чтобы в ней появились бинарные переменные, а потом «превращаем» их в переменные со значениями на отрезке  $[0, 1]$ .

Теперь легко перейти к обобщению  $k$ -means, надо лишь бинарные индикаторы принадлежности кластерам  $r_{it}$  «превратить» в вещественные из отрезка  $[0, 1]$ .

Вход алгоритма: выборка  $\{x_1, \dots, x_m\} \subseteq \mathbb{R}^n$ . Гиперпараметры алгоритма: натуральное число  $k$  (число кластеров для построения),  $\beta \in \mathbb{R}^+$ .

- Инициализация: выбираем случайно  $k$  центров кластеров

$$\{\mu_1, \dots, \mu_k\} \subseteq \mathbb{R}^n.$$

- Итерация алгоритма (итерации повторяются, пока центры кластеров не стабилизируются):
  - Для каждого  $i$ -го объекта вычисляем оценку принадлежности к каждому  $t$ -му кластеру:

$$r_{it} = \frac{\exp(-\beta \|x_i - \mu_t\|)}{\sum_j \exp(-\beta \|x_i - \mu_j\|)}.$$

- Пересчитываем центры кластеров:

$$\mu_t = \frac{1}{\sum_{i=1}^m r_{it}} \sum_{i=1}^m r_{it} x_i.$$

Мы легко на базе идеи  $k$ -средних получили нечёткую кластеризацию, иногда предложенный метод даже лучше (быстрее сходится, более адекватные на вид результаты) классического  $k$ -means, но

- необходимо выбирать значение гиперпараметра  $\beta \in \mathbb{R}^+$  (он отвечает за «резкость» степени принадлежности к кластерам), его выбор не так интуитивен как у числа кластеров  $k$ , кроме того, в задачах обучения без меток нет отложенного контроля, чтобы настраивать гиперпараметры),
- остались проблемы с кластеризацией «продолговатых» и разномошных кластеров.

В задачах без меток гиперпараметры часто задаются экспертно.

## Распространение близости (Affinity propagation)<sup>1</sup>

Пусть  $s(i, j)$  – близость (similarity) между точками  $x_i$  и  $x_j$ . В этом методе каждый кластер образуется своим лидером, который является объектом выборки. На базе исходных близостей для каждой пары точек выборки итерационно пересчитываются  $r(i, k)$  – **ответственность (responsibility)** и  $a(i, k)$  – **доступность (availability)**:

$$r(i, k) \leftarrow s(i, k) - \max\{a(i, k') + s(i, k') \mid k' \neq k\}$$

(ответственность формализует идею, что  $k$ -я точка может быть лидером для  $i$ -й, если они достаточно похожи),

$$a(i, k) \leftarrow \min\left(0, r(k, k) + \sum_{i' \notin \{i, k\}} r(i', k)\right)$$

( $i$ -я точка признаёт в  $k$ -й лидера, если  $k$ -я точка является лидером для многих). Обычно пересчитывают «с инерцией», например

$$r^{(t+1)}(i, k) = \lambda r^{(t)}(i, k) + (1 - \lambda) r^{\text{new}}(i, k).$$

После сходимости каждый объект  $x_i$  приписывается к кластеру с лидером  $x_j$ , для которого выражение  $a(i, j) + r(i, j)$  максимально., т.е. образуются кластеры

$$C_t = \{i \mid t = \arg \max(a(i, \bullet) + r(i, \bullet))\},$$

потенциально каждая точка может образовать кластер, но на практике их получается меньше, см. рис. XX.11.

У описанного метода **высокая сложность**:  $O(m^2 n_{\text{iter}})$  и большие **требования к памяти**  $O(m^2)$ , т.к. мы постоянно вычисляем и используем  $m \times m$ -матрицу близостей. Зато нет необходимости экспертно задавать число кластеров.

<sup>1</sup> Brendan J. Frey, Delbert Dueck «Clustering by Passing Messages Between Data Points» Science Feb. 2007 <https://www.icmla-conference.org/icmla07/FreyDueckScience07.pdf>, есть также хорошее объяснение <https://habr.com/ru/post/321216/>

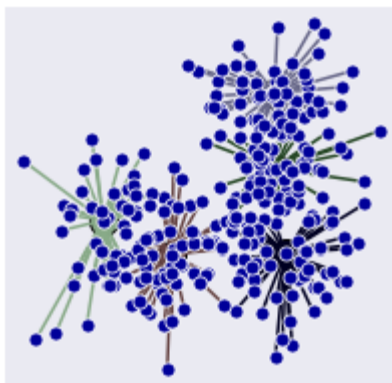


Рис. XX.11. Точки выборки соединены отрезками со своими лидерами.

### Сдвиг среднего (Mean Shift)

Этот метод прежде всего нацелен на обнаружение мод плотности<sup>1</sup>. Есть модификации метода, в которых выписывается оценка плотности по Парзену

$$\rho(x) \propto \sum_{i=1}^m K(x - x_i)$$

и далее применяется градиентный метод

$$x \leftarrow x + \eta \nabla \rho(x),$$

который стартует из всех точек выборки для обнаружения локальных максимумов плотности, т.е. мод. В классическом алгоритме для каждой точки выборки  $x$  рассматривается лишь её окрестность  $N(x)$ ,  $N(x) = \{x_i \mid \rho(x, x_i) \leq d\}$ , задаваемая гиперпараметром – радиусом  $d$ , (можно использовать и альтернативные формализации понятия окрестность) и оценивается «центр масс» в этой окрестности:

$$m(x) = \frac{\sum_{x_i \in N(x)} K(x - x_i) x_i}{\sum_{x_i \in N(x)} K(x - x_i)},$$

где  $K(z) = \exp(-\|z\|_2^2 / h)$  – функция ядра с гиперпараметром «ширина»  $h$  (здесь можно использовать и альтернативные формализации ядра), затем точка сдвигается в сторону центра:

<sup>1</sup> Fukunaga K., Hostetler L. The estimation of the gradient of a density function, with applications in pattern recognition //IEEE Transactions on information theory. – 1975. – Т. 21. – №. 1. – С. 32-40.



$$x \leftarrow x + \eta(m(x) - x),$$

$\eta \in (0,1)$  – гиперпараметр «темп сдвига». На рис. XX.12 показано несколько итераций алгоритма Mean Shift: изменение положений объектов в признаковом пространстве.

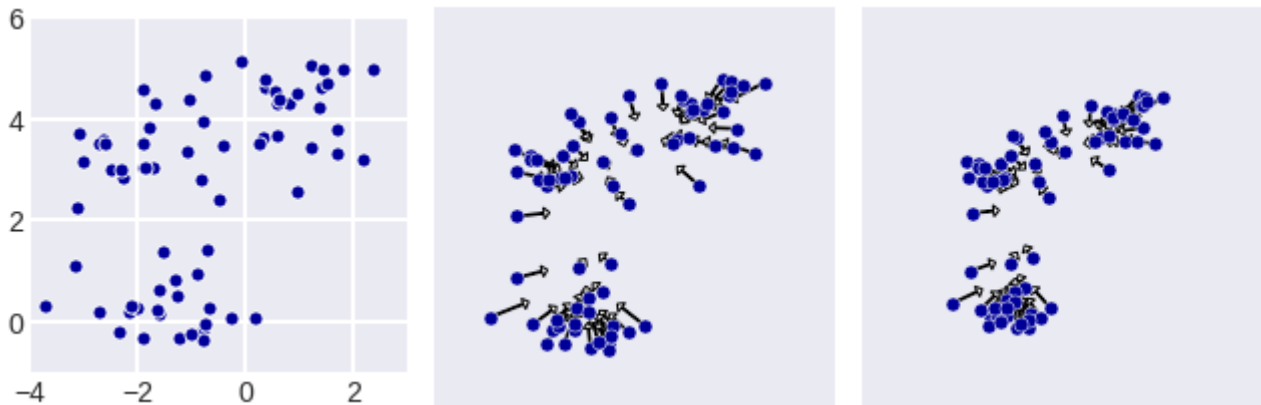


Рис. XX.12. Исходная выборка (слева) и две итерации (по центру и справа) метода Mean Shift.

Идея описанного сдвига – точки стремятся в центры «сгустка», итерации повторяются, пока точки не притянутся к этим центрам (с  $\varepsilon$ -точностью), каждый полученный центр и будет модой плотности, а соответствующий сгусток – кластером.

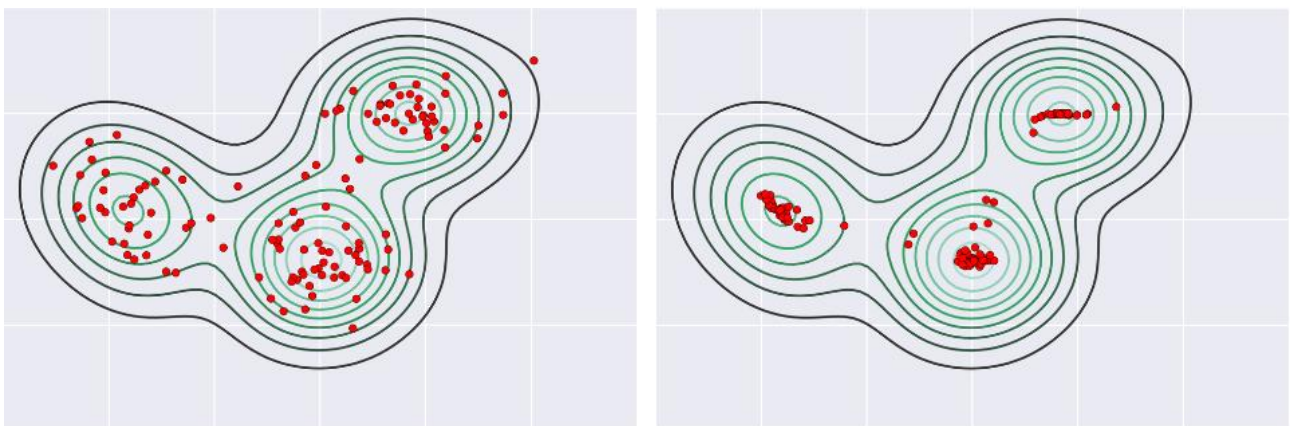


Рис. XX.13. Исходная (слева) конфигурация точек и итоговая (справа) в результате выполнения Mean Shift<sup>1</sup>.

В описанном методе достаточно много гиперпараметров:  $\varepsilon$ ,  $d$  и  $h$ , значения которых влияют на итоговое число кластеров. Как видно на рис. XX.13 некоторые точки медленнее подтягиваются к сгусткам, их можно детектировать, объявлять выбросами и удалять из выборки.

<sup>1</sup> Рисунок взят по адресу [https://github.com/mattnedrich/MeanShift\\_py](https://github.com/mattnedrich/MeanShift_py).

## Иерархическая кластеризация (Hierarchical clustering)

В отличие от плоской (Flat Clustering) иерархическая кластеризация получает серию вложенных друг в друга кластеризаций, которая изображается в виде дендрограммы, см. рис. XX.14: объекты разбиваются на два кластера (синяя часть), первый и второй разбираются рекурсивно (зелёная и красная часть) до тех пор, пока каждый объект не окажется в отдельном кластере. Есть два стандартных подхода для иерархической кластеризации:

- **Агломеративный (Agglomerative / bottom-up)**
  - дендрограмма строится снизу-вверх,
  - изначально число кластеров совпадает с числом объектов: каждый кластер содержит один объект обучения,
  - на каждом шаге объединяем два наиболее похожих (это надо формализовать) кластера,
  - останавливаемся, когда остаётся один кластер (он содержит все объекты) – дендрограмма построена.
- **Дивизионный (Divisive / top-down)**
  - дендрограмма строится сверху-вниз,
  - изначально есть один кластер, содержащий все объекты обучения,
  - кластер с максимальными внутрикластерными расстояниями (это также надо формализовать) расщепляется на два,
  - останавливаемся, когда число кластеров совпадает с числом объектов – дендрограмма построена.

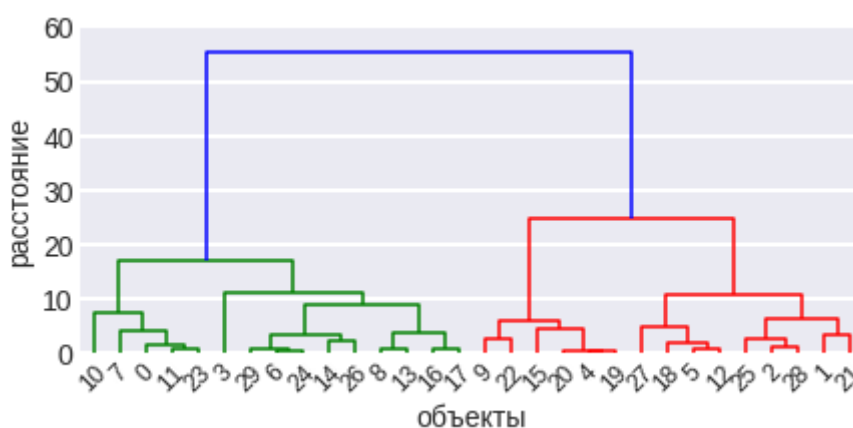


Рис. XX.14. Дендрограмма.

Иерархическая кластеризация формально не требует задания числа кластеров, но при необходимости построение дендрограммы можно остановить при достижении этого числа. В этом контексте подход сверху-вниз более предпочтителен, так как при небольшом числе кластеров не выполняются лишние действия. Однако **на практике чаще используют и проще реализовать подход снизу-вверх**, т.к. объединять кластеры проще, чем разделять.

Действие с более простой и понятной реализацией определяет популярность подхода.

На рис. 15 показана выборка в модельной задаче, очевидно, что точки с номерами 2 и 3 находятся ближе друг к другу, в подходе снизу-вверх их кластеры {2} и {3} объединяются первыми, на дендрограмме это обозначается слиянием линий из точек 2 и 3, по горизонтали момент слияния соответствует расстоянию между точками. Аналогично и другие слияния отображены на дендрограмме. В данном случае расстояние между кластерами определялось как расстояние между двумя ближайшими их представителями. Например, если продолжать сливать (объединять) похожие (ближайшие) кластеры, то далее объединяются {4} и {10}, потом {1} и {9}, потом {4, 10} и {6}, а на дендрограмме уровень объединения соответствует расстоянию между точками 4 и 6 (т.к. точка 4 ближе к точке 6, чем точка 10).

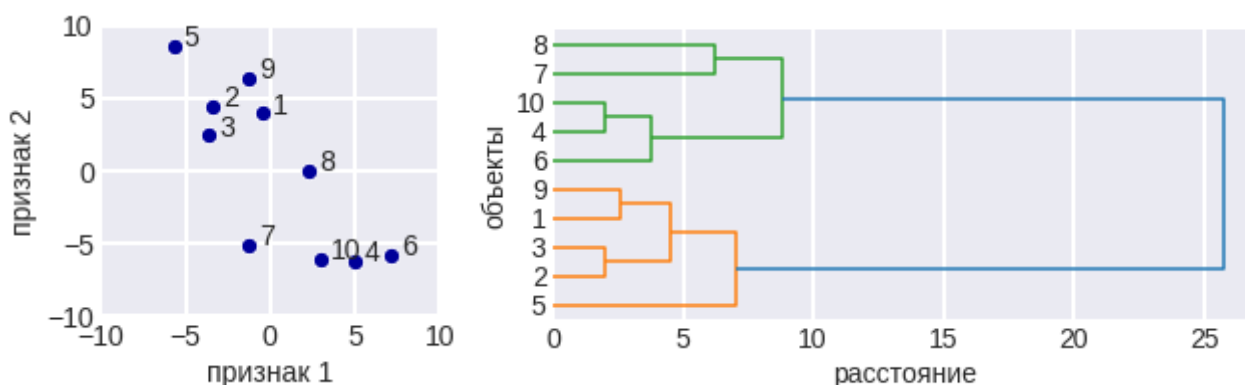


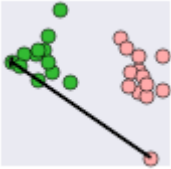

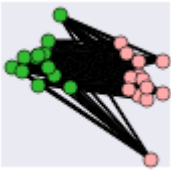
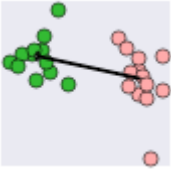
Рис. XX.15. Выборка (слева) и дендрограмма (справа).

Как мы видим, в подходе снизу-вверх необходимо уметь вычислять расстояние<sup>1</sup> между кластерами, есть следующие стандартные способы (см. табл., типы способов называются Linkage): **Complete, Single, Average, Centroid**. Каждый из способов имеет физическую интерпретацию. При измерении расстояния между столбом и стеной рулеткой соединяют ближайшие точки (такая же

Альтернативы появляются из-за разного физического смысла различных формализаций.

<sup>1</sup> Формально перечисленные ниже формализации не всегда являются расстояниями (не выполняются некоторые требования, например аксиома треугольника).

логика в Single Linkage), расстоянием между двумя городами чаще считают расстояние между их центрами или какими-то выделенными пунктами, например «нулевыми километрами» (Centroid Linkage). Матожидание расстояния между представителями кластеров также очень логично использовать, у нас кластеры являются конечными множествами и оценка такого матожидания – усреднение всех попарных расстояний между представителями двух кластеров (Average Linkage). Максимум попарных расстояний также является довольно удобной статистикой (Complete Linkage). Если представить, что курьер едет из города А в город В, а его местонахождение и место назначения не известны, то в самом худшем случае ему придётся преодолеть путь между парой максимально удалённых точек двух городов.

Linkage		Описание
Complete		Максимальное межкластерное расстояние $\max_{x \in A, x' \in B} \text{dis}(x, x')$
Single		Минимальное межкластерное расстояние $\min_{x \in A, x' \in B} \text{dis}(x, x')$
Average		Среднее межкластерное расстояние $\frac{1}{ A  \cdot  B } \sum_{x \in A, x' \in B} \text{dis}(x, x')$
Centroid		Расстояние между центроидами кластеров $\text{dis}(\bar{A}, \bar{B})$

Предложим ещё одно вполне естественное **расстояние Варда (Ward's measure)**. Если ввести ненормированный разброс  $\lambda(C)$  кластера  $C$  как

$$\lambda(C) = \sum_{x \in X} \|x - \bar{C}\|^2,$$

где  $\bar{C}$  – центр кластера, то нетрудно доказать, что после объединения двух кластеров этот разброс изменится следующим образом:

$$\lambda(A \cup B) - \lambda(A) - \lambda(B) = \frac{1}{\frac{1}{|A|} + \frac{1}{|B|}} \|\bar{A} - \bar{B}\|^2$$

(из разброса объединения вычитаем сумму разбросов объединяемых кластеров), это выражение и является «расстоянием Варда» (нормированный квадрат расстояния между их центрами). Заметим, что *inertia* в (XX.1) является суммой ненормированных разбросов кластеров.

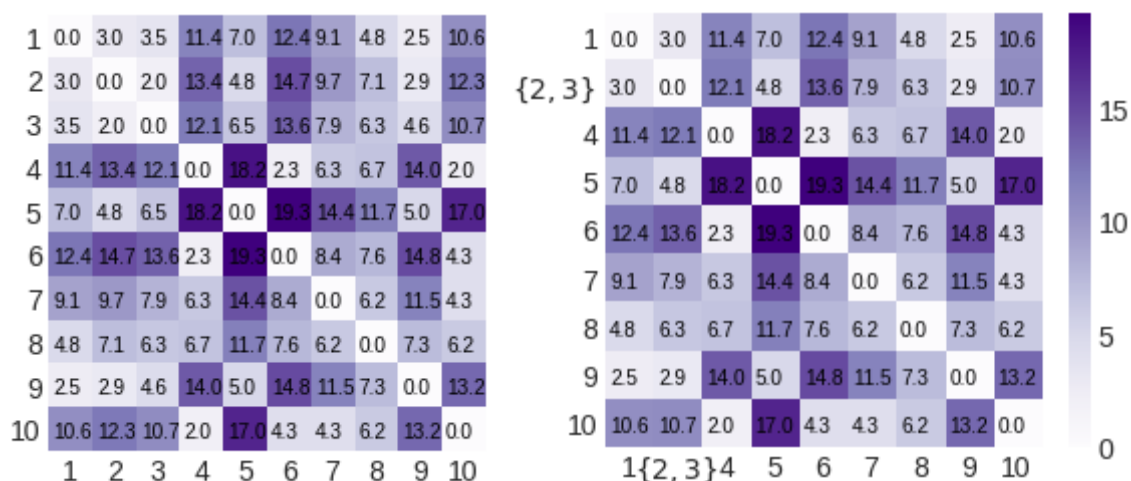


Рис. XX.16. Матрицы попарных расстояний на двух итерациях Single Link.

На рис. XX.16 показано первое объединение в методе Single Link, который мы иллюстрировали на рис. XX.15. В матрице попарных расстояний между кластерами (слева) – на первом шаге она совпадает с матрицей попарных расстояний между точками – мы находим минимальный недиагональный элемент, здесь в позиции (2, 3). Это означает, что объединяются соответствующие кластеры: {2} и {3}. Для этого мы берём поэлементный минимум 2-й и 3-й строки, записываем полученный результат во 2-ю строку и 2-й столбец, а 3-ю строку и 3-й столбец удаляем. В полученной матрице (рис. XX.16 справа) расстояния изменились только в строке и столбце, которые соответствуют как раз расстояниям до нового кластера, а расстояние до нового кластера равно минимуму расстояний до его подкластеров. Это процесс итеративно продолжается. Скажем, при реализации CompleteLink пришлось бы брать поэлементный максимум.

На рис. XX.17 показаны результаты на модельных задачах при использовании разных способов вычисления расстояний между кластерами в иерархической кластеризации (отметим, что во всех задачах кластеризация проводилась на экспертно заданное «правильное» число кластеров, кроме последней, в ней точки с равномерным распределением в квадрате и число кластеров равно 3).



Метод **Single Linkage**, в отличие от других методов, прекрасно справляется с достаточно разнесёнными точечными конфигурациями произвольной формы, например вложенными окружностями (т.к. он итерационно близкие точки «кладёт» в один кластер). Но стоит конфигурации стать достаточно шумной (4я задача), как он даёт не совсем адекватный результат. Впрочем, в таких ситуациях, хороши видны аномалии – маленькие кластеры, выделенные методом Single Linkage (см. конфигурации 2, 4 и 5).

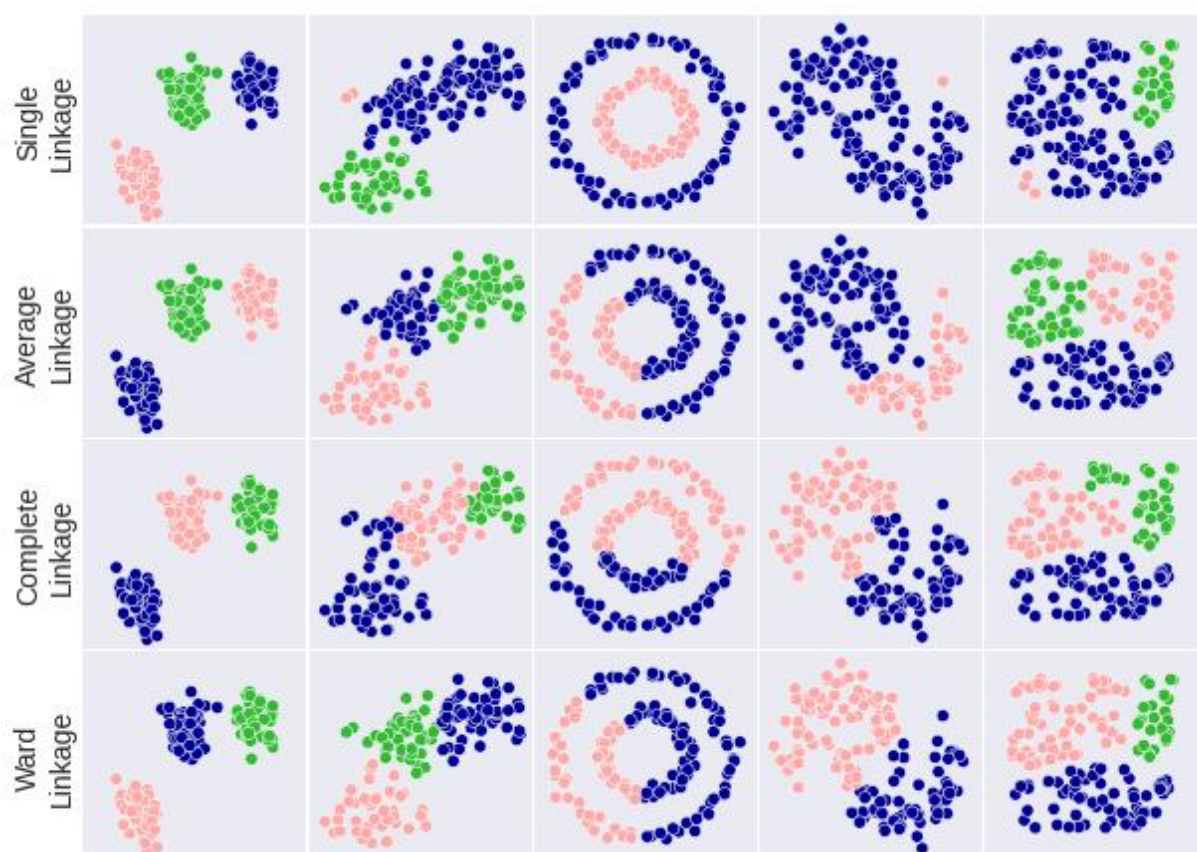


Рис. XX.17. Сравнение иерархической кластеризации при разных расстояниях.

### Использование графа (или матрицы связности) при кластеризации

Алгоритмы кластеризации можно сделать более универсальными – чтобы они учитывали особенности многообразия, на котором располагаются точки нашей выборки. На рис. XX.18 (слева) показана обычная кластеризация (в данном случае одна из полученных с помощью алгоритма иерархической кластеризации). Все точки выборки находятся около поверхности «скрученного листа». Точки на разных, но близких участках листа попадают в один кластер. Нам бы хотелось, чтобы кластеризация просто делила лист на участки, см. рис. XX.18 (справа). Чтобы добиться такого результата, при кластеризации надо учитывать топологию системы точек. Для этого многие реализации



кластеризации допускают использование в качестве гиперпараметра  $m \times m$ -матрицы связности<sup>1</sup> (connectivity), где  $m$  – число объектов. Такая матрица может быть матрицей смежности графа, который представляет соседство объектов, например это граф  $k$ -соседства (каждая точка соединяется рёбрами с  $k$  ближайшими точками выборки). При использовании такого графа в иерархической кластеризации получаем результат, показанный на рис. XX.18 (справа), поскольку теперь в агломеративном подходе кластеры являются вершинами графа и объединяться могут только соседние вершины-кластеры (при этом эта пара вершин заменяется на одну с пересчётом соседства).

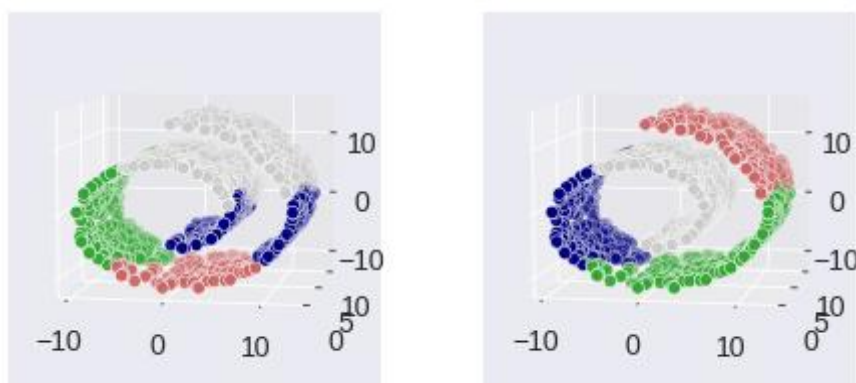


Рис. XX.18. Пример обычной кластеризации (слева) и с матрицей соседства (справа).

Опишем метод **спектральной кластеризации (Spectral Clustering)**, который также использует граф для кластеризации, хорош для точечных конфигураций типа «вложенные окружности» и берёт начало из теории графов<sup>2</sup>. Изложим алгоритм по пунктам.

1. На первом этапе по данным  $\{x_1, \dots, x_m\}$  строится граф  $G=(V, E)$   $k$ -соседства с весами, множество вершин соответствует исходным объектам, для простоты можно обозначить их номерами объектов:  $V=\{1, 2, \dots, m\}$ , вершина  $i$  соединена с номерами вершин, которые соответствуют ближайшим  $k$  объектам к объекту  $x_i$ . Ребро  $(i, j) \in E$  имеет вес  $w_{ij} = \exp(-\|x_i - x_j\|^2 / \sigma)$ ,  $\sigma$  – гиперпараметр метода. Для удобства считаем, что  $w_{ij} = 0$  при  $(i, j) \notin E$ , тогда все весам можно записать в матрицу  $W = \|w_{ij}\|_{m \times m}$ . На рис. XX.19 показан такой граф  $k$ -соседства (веса не отображаются).

<sup>1</sup> Например, иерархическая в sklearn, также задать функцию связности (от пар индексов точек).

<sup>2</sup> Полное описание есть здесь: Ulrike von Luxburg «A Tutorial on Spectral Clustering», 2007 <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.165.9323>

- Далее строится матрица Лапласа  $L = D - W$  (или нормированная матрица Лапласа  $L' = I - D^{-1}W$ ), где в диагональной матрице  $D$   $ii$ -й элемент равен сумме  $i$ -го столбца (и строки) матрицы  $W$ .
- Находятся собственные векторы, которые соответствуют минимальным собственным значениям матрица Лапласа в количестве  $d + 1$ :

$$v_1 = \tilde{1}, v_2, \dots, v_d$$

(первый вектор состоит из одних единиц и соответствует нулевому значению, т.к.  $L\tilde{1} = \tilde{0}$ ).

- Строится новая признаковая  $m \times d$ -матрица  $V = [v_2, \dots, v_d]$ , на которой применяется какой-нибудь известный алгоритм кластеризации (часто используют  $k$ -средних<sup>1</sup> из-за быстроты и простоты).

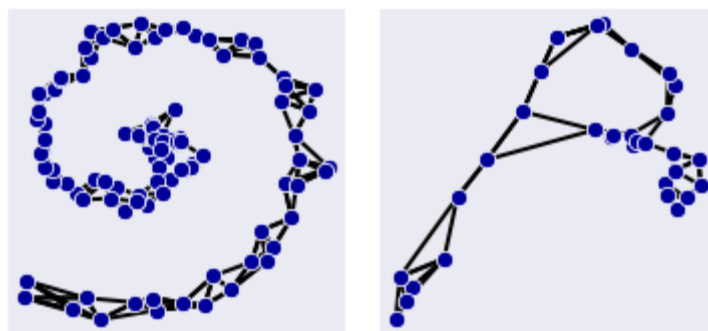


Рис. XX.19. Графы  $k$ -соседства точечных конфигураций<sup>2</sup>,  $k = 5$  (слева) и  $k = 4$  (справа).

Обратим внимание, что при спектральной (и выше при иерархической) кластеризации **можно использовать любой граф – не обязательно граф  $k$ -соседства**, с помощью графа в кластеризатор передаётся структура близости точечной конфигурации (чем ближе вес ребра к 1, тем ближе точки). Матрицу  $W$  называют также **матрицей сходства**, описанную кластеризацию рекомендуют применять в случае разреженной матрицы сходства. В случае  $k$ -соседства кластеризатор учитывает близость соседних точек, а близость остальных неявно определяется по графу.

Надо хорошо понимать, как на алгоритм влияет дополнительная информация, в данном случае, представленная в виде графа. И использовать это на практике.

<sup>1</sup> Здесь при определении  $k$ -соседства и в методе  $k$ -средних, вообще говоря, разные значения гиперпараметра  $k$ .

<sup>2</sup> В реализации `sklearn.neighbors.kneighbors_graph` при указании параметра  $k$  (`n_neighbors`) сама точка также считается своим соседом.

В описании алгоритма при определении весов у нас возник гиперпараметр  $\sigma$ , который сложно настраивать, но сам способ определения весов добавляет «лишнюю степень свободы» в метод, можно использовать произвольную функцию  $K(x_i, x_j)$ , например ядро.

## Графовые методы

Графовыми методами кластеризации называются методы, в которых используются (явно или неявно) графы. Изученные ранее SingleLink и CompleteLink также приписывают к графовым, ниже будет понятно почему. Пусть дана выборка  $\{x_1, \dots, x_m\}$  точек в метрическом пространстве с метрикой  $\rho$ , **пороговым графом с порогом  $D$**  называется граф с множеством вершин  $\{1, 2, \dots, m\}$  и множество рёбер  $V$ :

$$(i, j) \in V \Leftrightarrow \rho(x_i, x_j) \leq D.$$

Другими словами, в таком графе рёбрами соединены близкие с точностью до  $D$  объекты<sup>1</sup>. На рис. XX.20 показаны пороговые графы при разных значениях значения порога. В графовом методе SingleLink каждая связная компонента порогового графа считается кластером, а иерархическая кластеризация создаётся при изменении порога. Нетрудно видеть, что такая иерархическая кластеризация эквивалентна описанной ранее, которую мы также называли Single Linkage. На рис. XX.20 при пороге  $D=1$  две компоненты, а при пороге  $D \geq \sqrt{2}$  одна компонента связности.

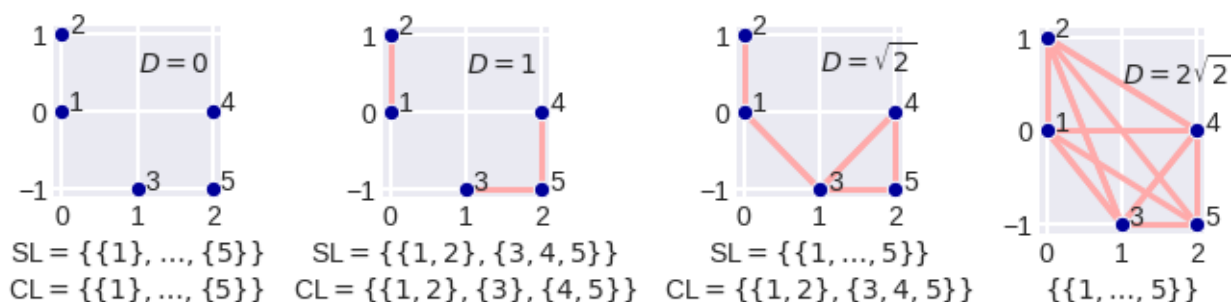


Рис. XX.20. Эволюция порогового графа.

В графовом методе CompleteLink каждая клика (полный подграф) считается кластером. Получается, что кластеры не определены однозначно, например при

<sup>1</sup> Часто, допуская вольность, отождествляем объекты и их номера.

$D=1$  один из кластеров  $\{1,2\}$ , а вторая связная компонента двумя способами распадается на клики:

$$\{3\} \cup \{4, 5\} \text{ или } \{3, 4\} \cup \{5\}.$$

Такая ситуация называется «ничьёй», на практике возникает не так часто и разрешается произвольно (одну из возможных кластеризаций считаем верной).

Заметим, что графовый метод CompleteLink эквивалентен описанному ранее Complete Linkage, в котором кластеры  $\{1\}$  и  $\{2\}$  находятся на расстоянии 1, также как пары  $\{4\}$ ,  $\{5\}$  и  $\{3\}$ ,  $\{4\}$ . Но стоит одну из последних пар объединить в один кластер, например  $\{4, 5\}$ , как этот кластер становится «далёк» от  $\{3\}$ : расстояние равно  $\sqrt{2}$  (в методе SingleLink оно остаётся равным 1).

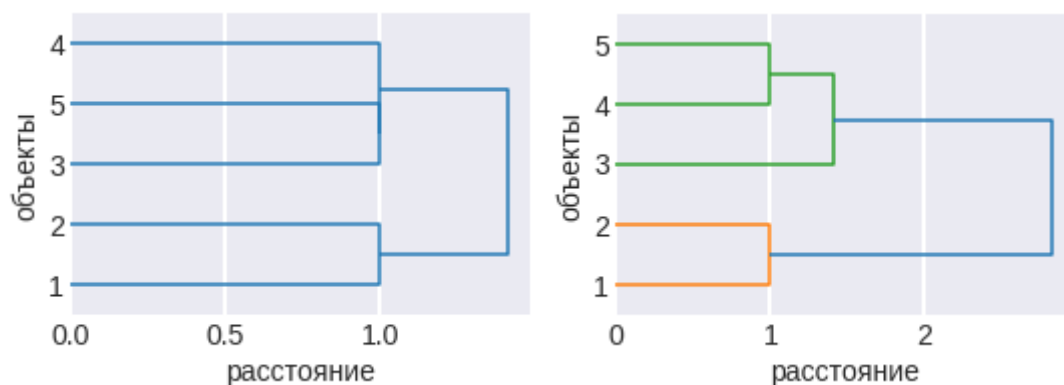


Рис. XX.21. Дендрограммы методов SingleLink (слева) и CompleteLink (справа).

Опишем также метод кластеризации на основе **минимального остовного дерева (MST, Minimum Spanning Tree)**. Если объекты выборки считать вершинами, то при соединении каких-то пар объектов отрезками, считая эти отрезки рёбрами, возникают различные графы. Если полученный граф будет деревом, то он называется **остовным деревом** (иными словами, это дерево, вершинами которого являются объекты выборки). **Минимальным остовным деревом** называется дерево сумма весов рёбер которого минимальна, под весом ребра понимается длина соответствующего отрезка (т.е. расстояние между соответствующими объектами).

Интересно, что минимальное остовное дерево может быть построено жадным алгоритмом. Сначала два ближайших объекта соединяем отрезками, потом итерационно добавляем ребро наименьшего веса, при добавлении которого граф остаётся лесом (объединением деревьев). Рёбра добавляются пока лес не превратится в дерево. На рис. XX.22 показаны минимальные остовные деревья, весом ребра тут считается значение евклидовой метрики между парой вершин этого ребра.

Метод кластеризации на основе MST следующий:

0. Построить минимально основное дерево.
1. Удалить из дерева  $(k - 1)$  ребро наибольшего веса ( $k$  – гиперпараметр, равный числу кластеров).
2. Компоненты связности полученного графа – кластеры.

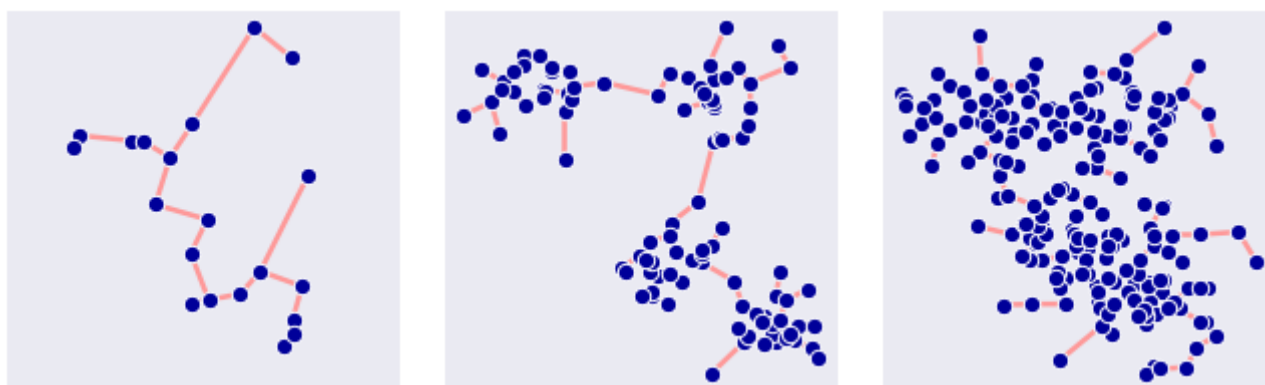


Рис. XX.22. Минимальные остовные деревья для разных конфигураций точек.

### DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Алгоритм кластеризации DBSCAN<sup>1</sup> по-простому можно назвать **алгоритмом с маркировкой объектов как шум, основанным на плотности**, его часто применяют на шумных данных при неизвестном априорно числе кластеров. Основные идеи: точки надо относить в один кластер, если они «достижимы по плотности» (density-reachable), т.е. есть довольно плотное связанное подмножество точек, которому они принадлежат; точку в очень разреженной области надо относить к шуму. Опишем алгоритм, считая, что каждый объект представляется точкой в признаковом пространстве:

Гиперпараметры алгоритма: радиус  $\varepsilon$  (eps, формализует отношение близости), порог соседства  $k$  (min\_samples, формализует плотность<sup>2</sup> точек).

1. Для каждой точки рассмотрим окрестность радиуса  $\varepsilon$ .
- 1.1. Если в окрестности есть как минимум  $k$  точек, то она называется

<sup>1</sup> Ester M. et al. A density-based algorithm for discovering clusters in large spatial databases with noise // kdd. – 1996. – Т. 96. – №. 34. – С. 226-231.

<sup>2</sup> Здесь слово «плотность» используется в бытовом смысле: «насыщенность».

**плотной / основной (core point).**

- 1.2. Иначе, если в окрестности есть плотные точки, то она называется **граничной / достижимой<sup>1</sup> (border point).**
- 1.3. Если точка не является плотной или граничной, то она считается **шумом (noise point).**
2. Множество основных точек кластеризуется известным алгоритмом кластеризации, чаще используют SingleLink, чтобы кластеры получались произвольных форм, при этом используют<sup>2</sup> пороговое расстояние  $\geq \varepsilon$ .
3. Граничные точки относят к тем кластерам, к плотным представителям которых они ближе.

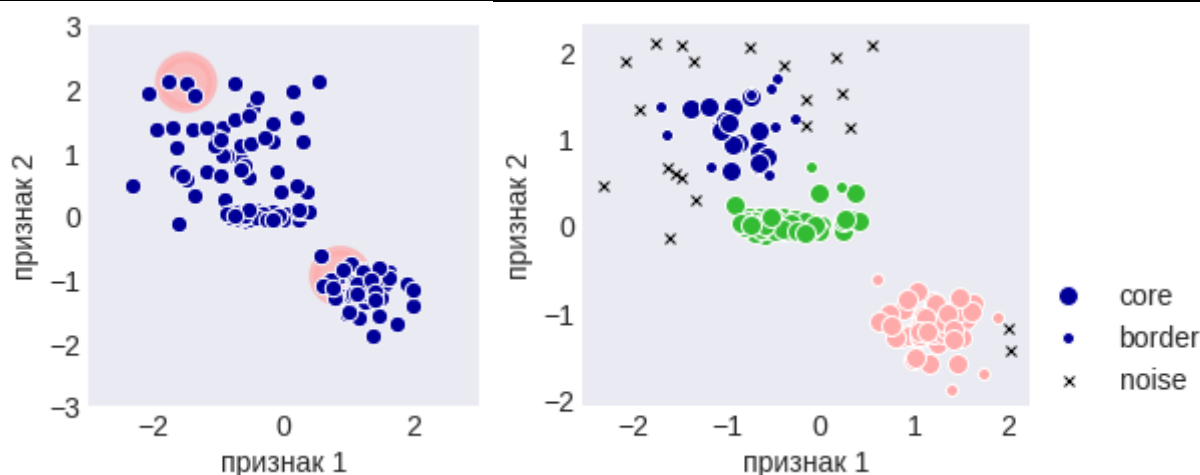


Рис. XX.23. Слева – конфигурация точек, справа – результат кластеризации.

На рис. XX.23 показана конфигурация точек, окрестности пары точек (слева, выделены розовым), в одной всего 3 точки, в другой – довольно много, а также результат кластеризации (справа). В алгоритме DBSCAN есть очевидные плюсы:

- кластеры получаются произвольной формы (например, на рис. XX.23 один из кластеров продолговатый),
- можно работать с большими данными, хотя надо поискать быстрые реализации самого алгоритма или его модификаций<sup>3</sup>,

<sup>1</sup> Иногда используют другое определение: точка достижима, если есть последовательность точек, которая начинается с данной точки, заканчивается плотной, и любые две соседние точки которой находятся на расстоянии  $< \varepsilon$ .

<sup>2</sup> Чаще здесь также используется расстояние  $\varepsilon$ .

<sup>3</sup> См. например, HDBSCAN в <https://github.com/scikit-learn-contrib>



- результат детерминированный (нет случайности, ответ однозначный), но в «быстрых» реализациях алгоритм может быть зависимостью от порядка объектов выборки, т.к. достижимость может определяться по наличию в окрестности точек с меткой «основная», а при простановке всех меток за один проход такой метки может не быть, если точка не была проанализирована).

Но также есть и очевидные минусы:

- неудобные гиперпараметры (число кластеров – довольно понятный параметр, но в DBSCAN используется близость  $\varepsilon$ , от которой это число и зависит, на рис. XX.24 показано, как меняется результат кластеризации при изменении гиперпараметра  $\varepsilon$ ),
- не подходит для точечных конфигураций с неоднородной плотностью (гиперпараметр  $k$  отвечает за «плотность» в окрестности точки, при этом в какой-то области может быть очень «густая» конфигурация точек, а в какой-то – очень разреженная, в идеале в таких областях надо использовать разное значение  $k$ ).

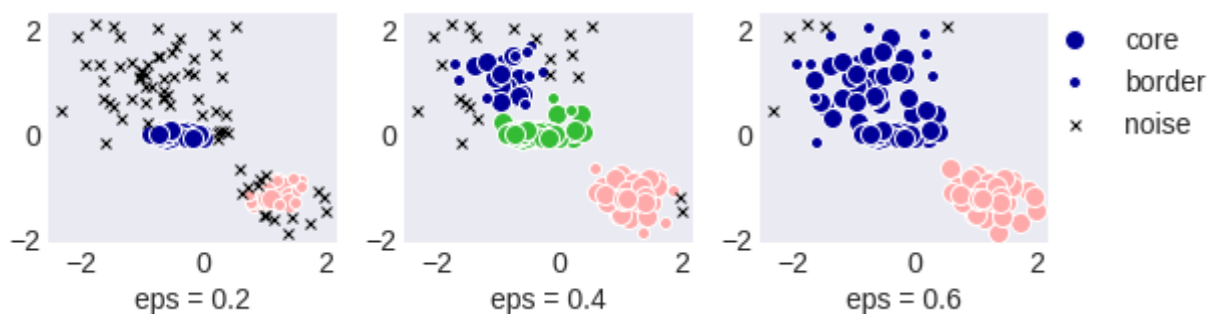


Рис. XX.24. Результаты кластеризации DBSCAN при разных значениях гиперпараметра  $\varepsilon$ .

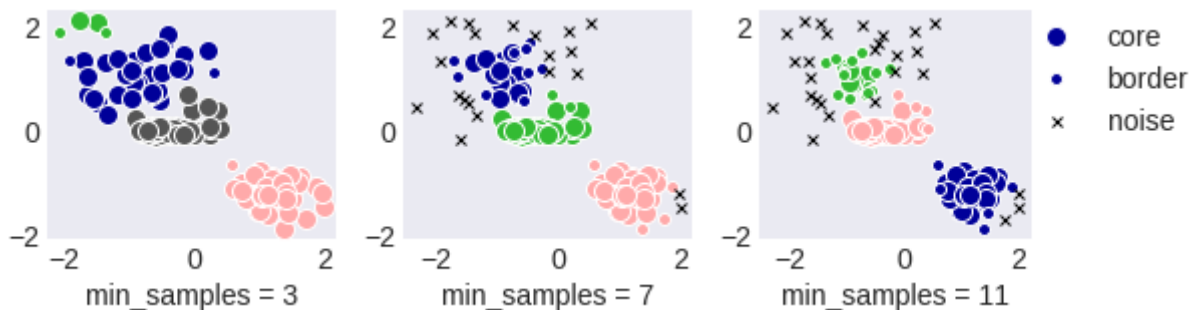


Рис. XX.25. Результаты кластеризации DBSCAN при разных значениях гиперпараметра  $k$ .

## Смесь гауссовских распределений (GMM – Gaussian Mixture Model)

Предположим, что наши данные распределены с плотностью, которая является выпуклой комбинацией плотностей нормальных распределений:

$$p(x) = \sum_{t=1}^k \pi_t \text{norm}(x | \mu_t, \Sigma_t),$$

$$\sum_{t=1}^k \pi_t = 1, \pi_t \geq 0.$$

Наша цель – «восстановить плотность по выборке», т.е. определить значения параметров  $\{\pi_t, \mu_t, \Sigma_t\}_{t=1}^k$  (коэффициенты в комбинации, векторы средних и матрицы ковариаций нормальных распределений). Указанная модель называется смесью **гауссиан** / **Gaussian Mixture Model (GMM)**<sup>1</sup> и является универсальным аппроксиматором плотности, т.е. любую плотность можно приблизить в таком виде, при этом может потребоваться достаточно много компонент смеси  $k$ . На практике используют небольшие значения  $k$ .

В контексте задачи кластеризации модель GMM реализует нечёткую кластеризацию на  $k$  кластеров. При применении метода максимального правдоподобия для определения параметров  $\{\pi_t, \mu_t, \Sigma_t\}_{t=1}^k$  получаем задачу

$$\sum_{i=1}^m \log \left( \sum_{t=1}^k \pi_t \text{norm}(x_i | \mu_t, \Sigma_t) \right) \rightarrow \max,$$

оптимизируемое выражение в общем случае довольно сложное (логарифм и экспонента в нормальном распределении сокращаются лишь при  $k=1$ ). Отметим также, что полученная задача оптимизации невыпуклая, решение неединственное (например, в нём есть инвариантность к перестановкам компонент). Для оптимизации можно применять методы градиентного спуска, с некоторыми трюками, например матрицы ковариации представляют в виде  $\Sigma_t = M_t M_t^T$  для обеспечения положительной определённости, но такой подход используют редко, кроме того, здесь остаются проблемы с эффективной оптимизацией<sup>2</sup>.

<sup>1</sup> Поиск параметров называют часто «разделением смеси гауссиан».

<sup>2</sup> См. подробнее в Hosseini R., Sra S. Manifold optimization for Gaussian mixture models //arXiv preprint arXiv:1506.07677. – 2015. <https://arxiv.org/pdf/1506.07677.pdf>

Предложим для обучения GMM следующий **ЕМ-алгоритм**<sup>1</sup>:

0. Случайно инициализируем параметры:  $\{\pi_t, \mu_t, \Sigma_t\}_{t=1}^k$ .

1. Повторять до сходимости.

1.1. (Е-шаг) По текущим параметрам вычислить:

$$\gamma_{it} = \frac{\pi_t \text{norm}(x_i | \mu_t, \Sigma_t)}{\sum_j \pi_j \text{norm}(x_i | \mu_j, \Sigma_j)}$$

(значение  $\gamma_{it}$  будем интерпретировать как вероятность того, что  $i$ -й объект лежит в  $t$ -м кластере).

1.2. (М-шаг) По параметрам  $\gamma_{it}$  пересчитать параметры кластеров:

$$m_t = \sum_{i=1}^m \gamma_{it} \text{ (назовём это объёмом } t\text{-го кластера),}$$

$$\mu_t = \frac{1}{m_t} \sum_{i=1}^m \gamma_{it} x_i \text{ (центр } t\text{-го кластера),}$$

$$\Sigma_t = \frac{1}{m_t} \sum_{i=1}^m \gamma_{it} (x_i - \mu_t)(x_i - \mu_t)^T \text{ («форма» } t\text{-го кластера),}$$

$$\pi_t = \frac{m_t}{m} \text{ (вероятность } t\text{-го кластера).}$$

На рис. XX.26 показаны начальные значения параметров (всех кроме вероятностей кластеров  $\pi_t$ ): векторы средних – розовыми точками, концентрические окружности (далее эллипсы) являются линиями уровня соответствующей плотности  $\text{norm}(x_i | \mu_t, \Sigma_t)$  с матрицей ковариаций  $\Sigma_t$ . На рис. XX.27 показано, как эти параметры меняются по итерациям: центры и формы кластеров «подстраиваются» под ближайшие точечные конфигурации, после 6й итерации наступает стабилизация параметров и правдоподобие не меняется, см. рис. XX.26.

<sup>1</sup> Изначально предложен в Dempster A. P., Laird N. M., Rubin D. B. Maximum likelihood from incomplete data via the EM algorithm //Journal of the royal statistical society: series B (methodological). – 1977. – Т. 39. – №. 1. – С. 1-22.

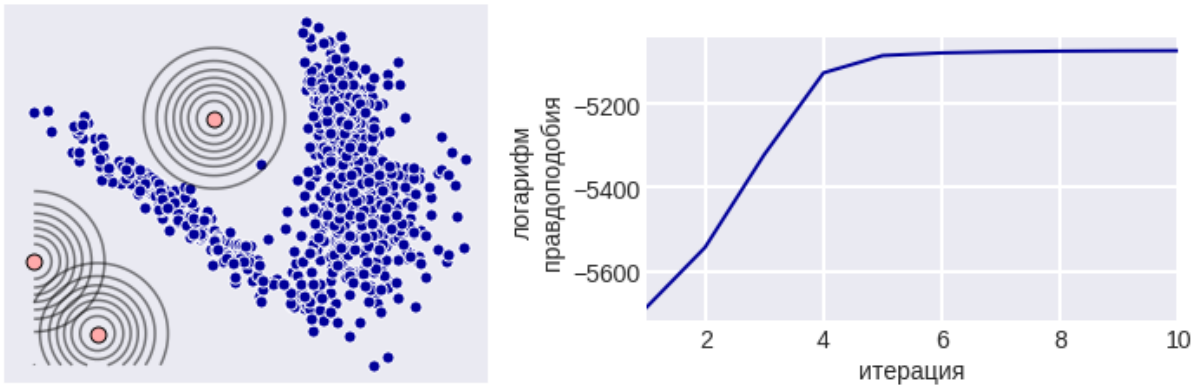


Рис. XX.26. Начальное приближение параметров (слева) и изменение логарифма правдоподобия по итерациям ЕМ-алгоритма (справа).

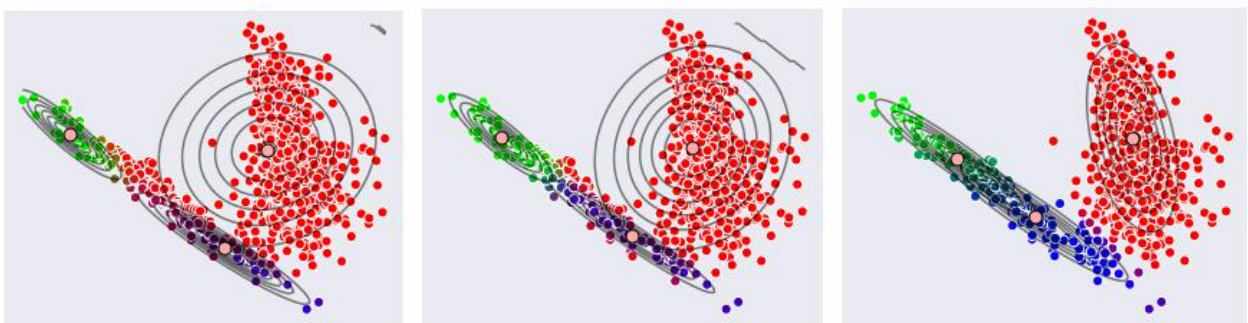


Рис. XX.27. Параметры смеси после 1й, 2й и 10й итерации ЕМ-алгоритма (слева направо).

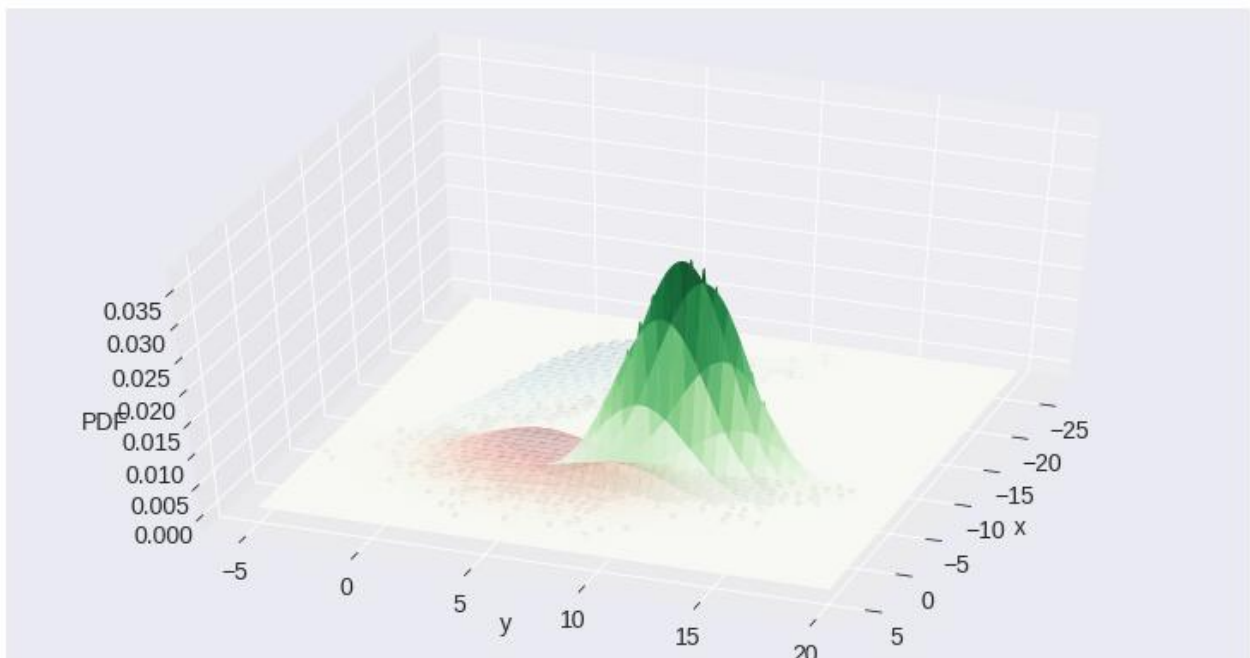


Рис. XX.28. Полученная выпуклая комбинация плотностей.

Результат ЕМ-алгоритма зависит от начальных приближений, на рис. XX.29 показаны итоговые положения компонент смеси при различных начальных инициализациях параметров. Их можно сравнить по значению правдоподобия,

на рис. XX.28 показана выпуклая смесь компонент, которой соответствует максимальное правдоподобие.

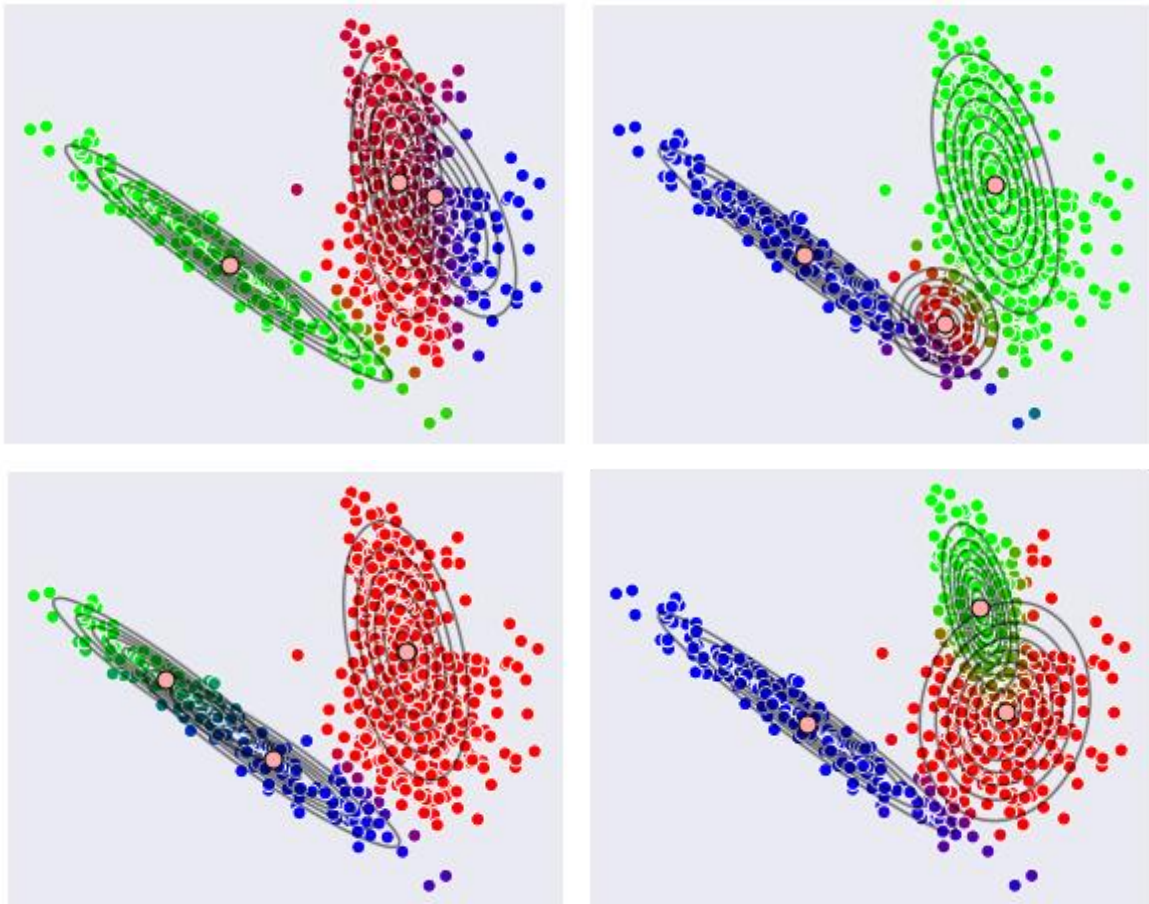


Рис. XX.29. Разные результаты EM-алгоритма.

Заметим, что в результате выполнения EM-алгоритма получаются также значения  $\gamma_{it}$ , которые мы интерпретировали как вероятности принадлежности к кластерам, поэтому **в результате решена задача нечёткой кластеризации**. Алгоритм похож на soft- $k$ -means, более того, он превращается в soft- $k$ -means, если

- распределения нормальные с одинаковыми ковариационными матрицами  $\Sigma_t = \varepsilon I$ ,
- кластеры равновероятны (equal priors):  $\pi_1 = \dots = \pi_k$ .

Если к тому же на E-шаге вместо нечёткого приписания объектов кластерам (вычисления  $\gamma_{it}$ ) делать чёткое приписание, то наш EM-алгоритм превращается в  $k$ -means. Преимущество GMM над  $k$ -means в том, что алгоритм  $k$ -means неявно предполагает, что классы являются «шарами примерно одного диаметра», теперь мы понимаем почему (при этих предположениях в него

превращается GMM), а GMM в общем случае описывает кластеры более сложной формы и разных размеров, см. рис. XX.30 (тут показано, что даже на шарообразных кластерах разного диаметра  $k$ -means проваливается, а GMM – нет). Алгоритм  $k$ -means можно использовать для инициализации центров в ЕМ-алгоритме<sup>1</sup>. Недостатком GMM в общем случае является слишком большое число параметров, поэтому используют технику **разделения параметров (parameter sharing)**, например все матрицы ковариаций считаются диагональными или равными<sup>2</sup>:

Разделение параметров часто используют в ML, особенно в нейросетях.

$$\Sigma_1 = \dots = \Sigma_k = \Sigma$$

(т.е. по сути обучается одна матрица  $\Sigma$  и все кластеры считаются одной формы).

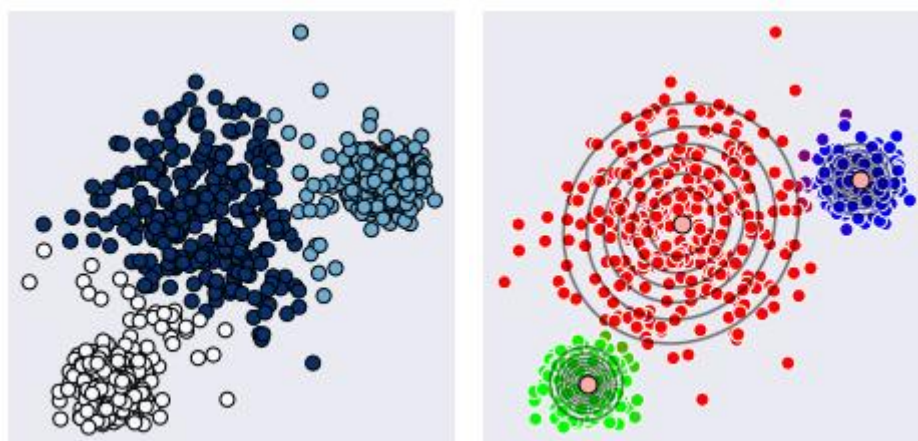


Рис. XX.30. Результат  $k$ -means (слева) и GMM/ЕМ (справа).

Изложенный ЕМ-алгоритм для решения GMM-задачи является:

- достаточно быстрым на практике,
- с понятной геометрией (в случае нормальных распределений неявно возникает расстояние Махалонобиса),
- естественно обобщает алгоритм  $k$ -means, но является более универсальным, подстраивается под более сложные формы и размеры кластеров.

<sup>1</sup> Так делается в scikit-learn, см. параметр `init_params`.

<sup>2</sup> И так делается в scikit-learn, см. параметр `covariance_type`.



При этом из очевидных недостатков следует отметить то, что

- мы фиксировали конкретный вид распределений в смеси (т.е. возможную форму кластеров)
- и число компонент смеси (т.е. число кластеров).

### Немного о выводе ЕМ-алгоритма<sup>1</sup>

Рассмотрим модель с латентными переменными Latent Variable Model. Пусть данные  $\{x_1, \dots, x_m\}$  порождаются следующим образом: при порождении каждой точки  $x_t$

1) сначала генерируется т.н. **латентная или скрытая переменная (hidden variable)**

$$z_t \sim p(z | \varphi),$$

здесь  $\varphi$  – параметры её распределения,

2) по латентной переменной генерируется **наблюдаемая переменная (observed variable)**

$$x_t \sim p(x | z_t, \theta),$$

аналогично  $\theta$  – параметры распределения.

В случае смеси произвольных распределений

$$p(x) = \sum_j \pi_j p(x | \theta_j), \quad \sum_{j=1}^k \pi_j = 1, \quad \pi_j \geq 0,$$

при генерации элемента выборки  $x_i$  сначала определяется латентный вектор  $(z_{i1}, \dots, z_{ik})$ , где

$$z_{ij} = I[x_i \sim p(x | \theta_j)] = \begin{cases} 1, & x_i \text{ из } j\text{-й компоненты,} \\ 0, & \text{иначе,} \end{cases}$$

т.е. это характеристический вектор номера компоненты. По формуле Байеса

<sup>1</sup> Подробнее см. в Bishop C. M. Pattern recognition and machine learning. – Springer, 2006.

$$\mathbf{P}(z_{ij}=1|x_i) = \frac{\mathbf{P}(z_{ij}=1) \cdot p(x_i | z_{ij}=1)}{\sum_t \mathbf{P}(z_{it}=1) \cdot p(x_i | z_{it}=1)} = \frac{\pi_j p(x_i | \theta_j)}{\sum_t \pi_t p(x_i | \theta_t)},$$

здесь пользуемся тем, что  $\pi_t = \mathbf{P}(z_{it}=1)$  при всех  $t \in \{1, 2, \dots, k\}$ . Введём обозначение  $\gamma_{ij} = \mathbf{P}(z_{ij}=1|x_i)$ . Мы только что обосновали Е-шаг в ЕМ-алгоритме, на нём происходит оценивание (estimation – отсюда буква Е в названии) вероятности, которую мы обозначили через  $\gamma_{ij}$ . Заметим, что при оценивании мы считаем известными (или уже оценёнными) параметры распределений  $\theta_t$ ,  $t \in \{1, 2, \dots, k\}$ .

Логарифм полного правдоподобия в нашей модели со скрытыми переменными запишется в виде

$$\log \prod_i \prod_j \pi_j^{z_{ij}} p(x_i | \theta_j)^{z_{ij}} = \sum_i \sum_j z_{ij} (\log \pi_j + \log p(x_i | \theta_j)),$$

теперь возьмём матожидание

$$\mathbf{E}_z \sum_i \sum_j z_{ij} (\log \pi_j + \log p(x_i | \theta_j)) = \sum_i \sum_j \gamma_{ij} (\log \pi_j + \log p(x_i | \theta_j)),$$

в распределении Бернулли матожидание совпадает с вероятностью:  $\mathbf{E}_z z_{ij} = \gamma_{ij}$ .

На М-шаге (от английского maximization – максимизация) производится оптимизация полученного «взвешенного правдоподобия»

$$J = \sum_i \sum_j \gamma_{ij} (\log \pi_j + \log p(x_i | \theta_j)) \rightarrow \max.$$

Если максимизировать по параметрам  $\pi_j$ , то это условная оптимизация, выпишем лагранжиан и продифференцируем его

$$\frac{\partial}{\partial \pi_t} \left[ J - \lambda \left( \sum_j \pi_j - 1 \right) \right] = \sum_i \frac{\gamma_{it}}{\pi_t} - \lambda = 0,$$

$$\pi_j = \sum_i \frac{\gamma_{ij}}{\lambda}.$$

Учитывая условия нормировки (сумма всех  $\pi_j$  равна 1), получаем

$$\pi_t = \frac{\sum_i \gamma_{it}}{\sum_j \sum_i \gamma_{ij}} = \frac{m_t}{m},$$

в старых обозначениях  $m_t = \sum_i \gamma_{it}$ . Эту формулу мы также использовали при описании ЕМ-алгоритма. До сих пор не было предположений относительно распределений компонент. Если использовать гауссианы, т.е.  $p(x_i | \theta_j) = \text{norm}(x_i | \mu_j, \Sigma_j)$ , тогда

$$J \propto \sum_i \sum_j \gamma_{ij} (x_i - \mu_j)^T \Sigma_j^{-1} (x_i - \mu_j),$$

дифференцируем (вычисляем градиент) по параметру  $\mu_j$  (тут безусловная оптимизация) и приравняем у нулю, получаем

$$\sum_i \gamma_{it} \Sigma_t^{-1} (x_i - \mu_t) = 0,$$

$$\sum_i \gamma_{it} x_i = \sum_i \gamma_{it} \mu_t,$$

$$\mu_t = \frac{\sum_i \gamma_{it} x_i}{\sum_i \gamma_{it}} = \frac{1}{m_t} \sum_i \gamma_{it} x_i.$$

Эту формулу использовали в ЕМ-алгоритме для пересчёта центров. Аналогично можно обосновать формулу для пересчёта матриц ковариаций. Вместо нормального распределения можно исследовать и другие, например для распределения Бернулли

$$J \propto \sum_i \sum_j \gamma_{ij} \log(\mu_j^{x_i} (1 - \mu_j)^{1-x_i}) = \sum_i \sum_j \gamma_{ij} (x_i \log \mu_j + (1 - x_i) \log(1 - \mu_j)),$$

после дифференцирования и приравнивания к нулю получаем

$$\sum_i \left( \gamma_{ij} \frac{x_i}{\mu_j} - \gamma_{ij} \frac{(1 - x_i)}{(1 - \mu_j)} \right) = 0,$$

$$\frac{\sum_i \gamma_{ij} x_i}{\mu_j} = \frac{\sum_i \gamma_{ij} x_i (1 - x_i)}{1 - \mu_j},$$

$$\mu_j = \frac{\sum_i \gamma_{ij} x_i}{\sum_i \gamma_{ij}} = \frac{1}{m_t} \sum_i \gamma_{ij} x_i,$$

формула аналогична формуле для нормального распределения.

Итак, ЕМ-алгоритм возможен для любых смесей распределений, в результате получаются оценки вероятностей (у них понятная интерпретация), шаги алгоритма организованы так, что не уменьшается правдоподобие<sup>1</sup>.

В заключение отметим, что задача максимизации правдоподобия в GMM не совсем правильно формализует желаемое решение. Кластер с маленькой дисперсией в центре точки выборки сколь угодно увеличивает правдоподобие, этот эффект называется сингулярностью, см. рис. XX.31.

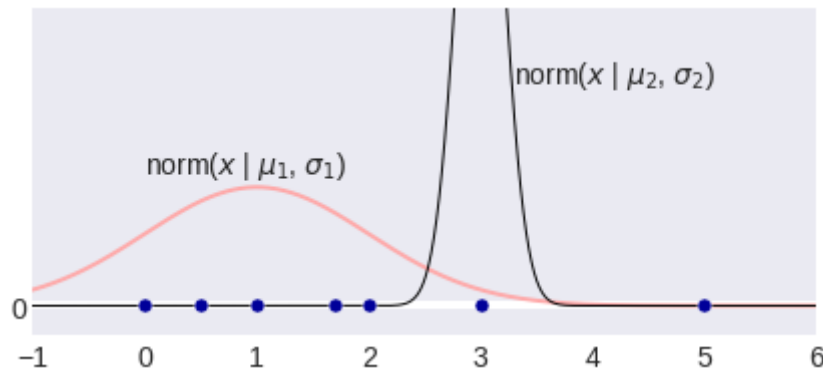


Рис. XX.31. Иллюстрация сингулярности.

### Оценка качества кластеризации

Если неизвестны правильные ответы кластеризации (а это типичный случай, в постановке задачи мы не предполагали, что объекты имеют какие-то метки), то используют т.н. **внутренние показатели (Internal Evaluation)**. Это эвристические формулы, которые часто оценивают малость внутрикластерных расстояний и разнесённость кластеров. Пусть множество объектов

$$X = \{x_1, \dots, x_m\},$$

оцениваемая чёткая кластеризация  $U = \{\{u_1\}, \dots, \{u_{|U|}\}\}$ :

<sup>1</sup> Мы это не аргументировали.

$X = u_1 \cup \dots \cup u_{|U|}$ ,  $u_i \cap u_j = \emptyset$  при  $1 \leq i < j \leq |U|$ . Наиболее популярные внутренние показатели, помимо упоминавшейся функции inertia (XX.1):

**Davies–Bouldin index<sup>1</sup>**

$$DB = \frac{1}{|U|} \sum_{i=1}^{|U|} \max_{j \neq i} \left( \frac{d(u_i) + d(u_j)}{d(u_i, u_j)} \right)$$

**Dunn index<sup>2</sup>**

$$D = \frac{\min_{1 \leq i < j \leq |U|} d(u_i, u_j)}{\max d(u_i)}$$

**Silhouette<sup>3</sup>**

$$\text{silhouette}(x_i) = \frac{d(x_i, u_2) - d(x_i, u_1)}{\max(d(x_i, u_2), d(x_i, u_1))}$$

**Calinski-Harabasz Index<sup>4</sup> (VRC, Variance Ratio Criterion)**

$$\frac{\text{trace} \left( \frac{1}{|U| - 1} \sum_{i=1}^{|U|} |U_i| (x - c_i)(x - c_i)^T \right)}{\text{trace} \left( \frac{1}{m - |U|} \sum_{i=1}^{|U|} \sum_{x \in U_i} (x - c_i)(x - c_i)^T \right)}$$

Здесь  $d(u_i)$  – показатель разброса внутри кластера (в Davies–Bouldin index для оценки разброса чаще используют стандартное отклонение, в Dunn index – максимальное попарное расстояние),  $d(u_i, u_j)$  – расстояние между  $i$ -м и  $j$ -м кластером, может вводиться по-разному (в Davies–Bouldin index чаще используют евклидово расстояние между центроидами),  $d(x_i, u_j)$  – расстояние между  $i$ -м объектом и  $j$ -м кластером, также может вводиться по-разному (в Silhouette чаще вычисляется как среднее расстояние до точек кластера),  $c_i$  – центроид кластера  $u_i$ . При вычислении Силуэта (Silhouette) считается, что  $x_i \in u_1$ ,  $d(x_i, u_2) \leq d(x_i, u_3) \leq \dots$ , т.е. кластеры занумерованы специальным образом. Таким образом, **силуэт точки оценивает, насколько точка ближе к своему кластеру, чем к ближайшему чужому** (силуэт может быть и отрицательным, например, точка на границе большого своего кластера может быть ближе к маленькому соседнему). Мы ввели силуэт для отдельного

<sup>1</sup> Davies D. L., Bouldin D. W. A cluster separation measure //IEEE transactions on pattern analysis and machine intelligence. – 1979. – №. 2. – С. 224-227.

<sup>2</sup> Dunn J. C. Well-separated clusters and optimal fuzzy partitions //Journal of cybernetics. – 1974. – Т. 4. – №. 1. – С. 95-104.

<sup>3</sup> Rousseeuw P. J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis //Journal of computational and applied mathematics. – 1987. – Т. 20. – С. 53-65.

<sup>4</sup> Caliński T., Harabasz J. A dendrite method for cluster analysis //Communications in Statistics-theory and Methods. – 1974. – Т. 3. – №. 1. – С. 1-27.

объекта, при оценивании качества кластеризации можно усреднить силуэты всех объектов.

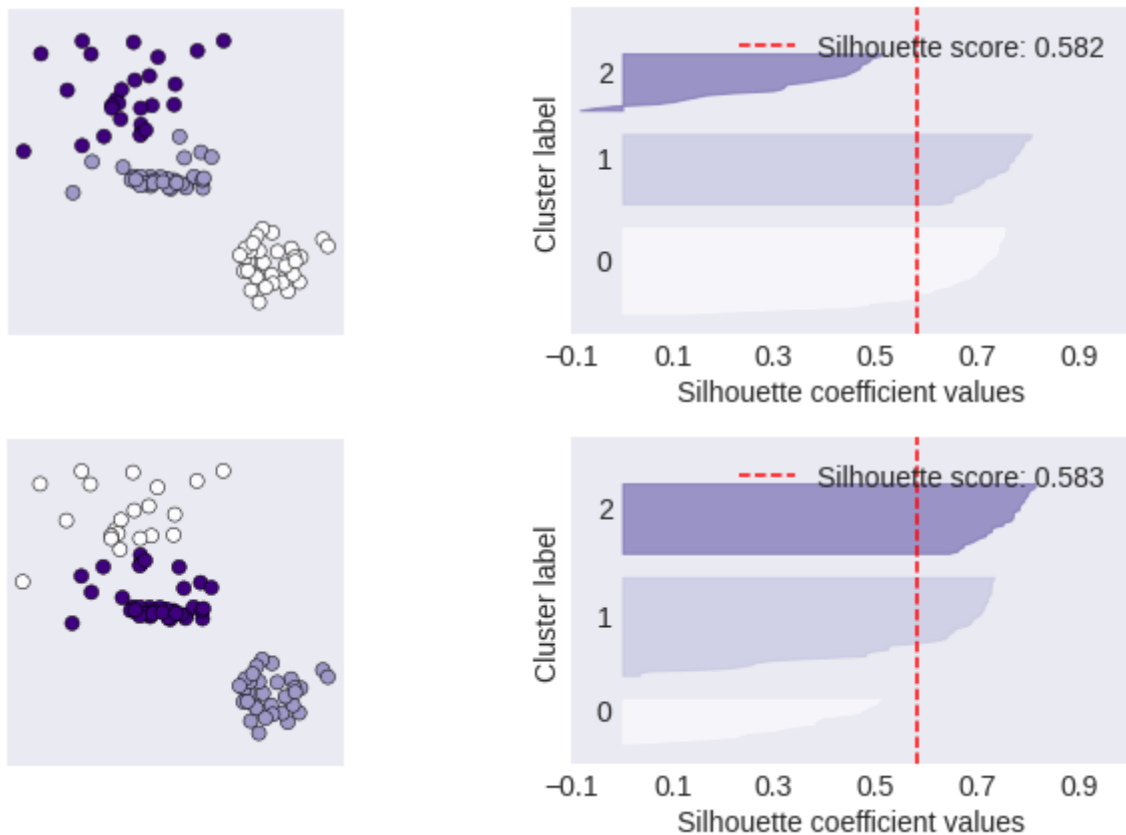


Рис. XX.32. Кластеризация (слева) и силуэты объектов (справа).

На рис. XX.32 показаны две разные кластеризации системы точек и силуэты<sup>1</sup> — их принято изображать в виде такой вертикальной гистограммы. В первой кластеризации есть отрицательные силуэты, но средние значения силуэтов (показаны красной пунктирной линией) двух кластеризаций не сильно отличаются.

Иногда бывает известна истинная кластеризация. Например, когда в задаче классификации исследуется признаковое пространство с целью понять, имеет ли оно кластерную структуру. Тогда исходное разбиение на классы можно считать истинной кластеризацией. В такой ситуации используются **внешние показатели (External Evaluation)**, которые сравнивают две кластеризации:

$$U = \{\{u_1\}, \dots, \{u_{|U|}\}\},$$

$$V = \{\{v_1\}, \dots, \{v_{|V|}\}\}$$

<sup>1</sup> Изображения силуэтов получены с помощью <https://sklearn-evaluation.ploomber.io>



на схожесть (считаем, что  $U$  – ответ кластеризатора,  $V$  – истинная кластеризация, если нам надо будет их различать). Каждая кластеризация порождает некоторую информационную энтропию, например для  $U$ :

$$H(U) = - \sum_{i=1}^{|U|} \frac{|u_i|}{m} \log \frac{|u_i|}{m}.$$

Для оценки кластеризации интуитивно должна подходить взаимная информация:

$$MI = \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} \frac{|u_i \cap v_j|}{m} \log \frac{\frac{|u_i \cap v_j|}{m}}{\frac{|u_i|}{m} \cdot \frac{|v_j|}{m}},$$

поскольку она показывает насколько определена одна кластеризация (разбиение на группы) при знании второй. На практике чаще используют **нормализованную взаимную информацию (Normalized Mutual Information)**:

$$NMI(U, V) = \frac{MI(U, V)}{\text{mean}(H(U), H(V))}$$

или

$$NMI(U, V) = \frac{MI(U, V)}{\sqrt{H(U) \cdot H(V)}},$$

а также **скорректированную взаимную информацию<sup>1</sup> (Adjusted Mutual Information)**:

$$AMI(U, V) = \frac{MI(U, V) - \mathbf{E}(MI(U, V))}{\max(H(U), H(V)) - \mathbf{E}(MI(U, V))},$$

здесь матожидание берётся по случайным кластеризациям выдаваемым кластеризатором, поэтому  $AMI(U, V) = 1$ , если кластеризации совпадают и  $AMI(U, V) = 0$ , если кластеризация-ответ случаен.

<sup>1</sup> Vinh N. X., Epps J., Bailey J. Information theoretic measures for clusterings comparison: is a correction for chance necessary? //Proceedings of the 26th annual international conference on machine learning. – 2009. – С. 1073-1080.

Один из самых популярных внешних показателей качества кластеризации – **V-мера** – **является средним гармоническим однородности (homogeneity) и полноты (completeness)**. Однородность оценивает, насколько верно, что каждый кластер содержит только объекты отдельного класса:

V-мера в кластеризации напоминает F-меру в классификации.

$$h(U, V) = \begin{cases} 1, & H(U, V) = 0, \\ 1 - H(U | V) / H(U), & H(U, V) \neq 0, \end{cases}$$

где

$$H(U | V) = - \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} \frac{|u_i \cap v_j|}{m} \log \frac{\frac{|u_i \cap v_j|}{m}}{\frac{|v_j|}{m}}.$$

Полнота оценивает, насколько верно, что все объекты конкретного класса отнесены в один кластер:

$$c(U, V) = \begin{cases} 1, & H(V, U) = 0, \\ 1 - H(V | U) / H(V), & H(V, U) \neq 0. \end{cases}$$

Искусственный способ максимизации полноты – выдать тривиальную кластеризацию  $U = \{X\}$  (отнести все объекты в один кластер). Нетрудно видеть, что  $c(U, V) = h(V, U)$ .

Также есть универсальный подход к оцениванию качества кластеризации при известных классах. Проблема сравнения двух кластеризаций в том, что оно должно учитывать инвариантность относительно перенумерации кластеров. Также качество не должно сильно меняться при незначительных изменениях: перенесении объекта из одного кластера в другой, расщеплении кластера на два и т.п. Заметим, что каждая кластеризация  $U$  порождает бинарные метки  $a_U \in \{0, 1\}$  на множестве пар

Приём: для оценки качества решения задачи сводим задачу к той, качество решения которой мы умеем оценивать.

$$M^2 = \{1, \dots, m\} \times \{1, \dots, m\} = \{(1, 1), \dots, (i, j), \dots, (m, m)\} :$$

$$a_U(i, j) = 1 \Leftrightarrow \exists u \in U : \{i, j\} \subseteq u,$$

т.е. пара номеров помечается меткой 1 тогда и только тогда, когда объекты с такими номерами лежат в одном кластере. Поэтому, сравнение кластеризаций

можно представить как сравнение чётких классификаций пар  $M^2$ . Например, значение ассигасы здесь имеет стандартное название **Rand index**<sup>1</sup>:

$$RI = \frac{TP + TN}{TP + FP + FN + TN},$$

здесь TP и прочие показатели вычисляются на множестве пар  $M^2$ . Кстати, каждая кластеризация задаёт отношение эквивалентности на множестве пар  $M^2$  (пара номеров в одном классе эквивалентности, если объекты с такими номерами в одном кластере), поэтому можно переписать Rand index в виде

$$RI = \frac{|\{i, j: (i \sim_U j) \& (i \sim_V j)\}| + |\{i, j: (i \not\sim_U j) \& (i \not\sim_V j)\}|}{C_m^2}.$$

Есть также скорректированная версия Adjusted Rand index:

$$ARI = \frac{RI - E[RI]}{\max(RI) - E[RI]}.$$

Задачу классификации пар использует также показатель Fowlkes-Mallows index (FMI), это среднее геометрическое точности и полноты для задачи классификации пар:

$$FMI = \frac{TP}{\sqrt{(TP + FP)(TP + FN)}}.$$

<sup>1</sup> Rand W. M. Objective criteria for the evaluation of clustering methods // Journal of the American Statistical association. – 1971. – Т. 66. – №. 336. – С. 846-850.

## Кластеризация: приложения

Приведём несколько примеров, которые заодно показывают причины применения кластеризации, помимо основных, которые обсуждались в начале главы.

**1. Кластеризация для интерпретации данных.** Если мы посмотрим на матрицу попарных расстояний объектов выборки, то скорее всего на ней невозможно будет заметить какие-то паттерны, см. рис. XX.33 – слева показана визуализация матрицы (значения её элементов определяют интенсивности закрашивания соответствующих ячеек на рисунке), но если объекты специальным образом упорядочить, то уже проще находить какие-то интересные артефакты, см. рис. XX.33 – справа объекты упорядочены по вхождению в кластеры: сначала идут объекты первого кластера, потом второго, потом третьего. При этом внутри кластера порядок случайный (это также логично изменить).

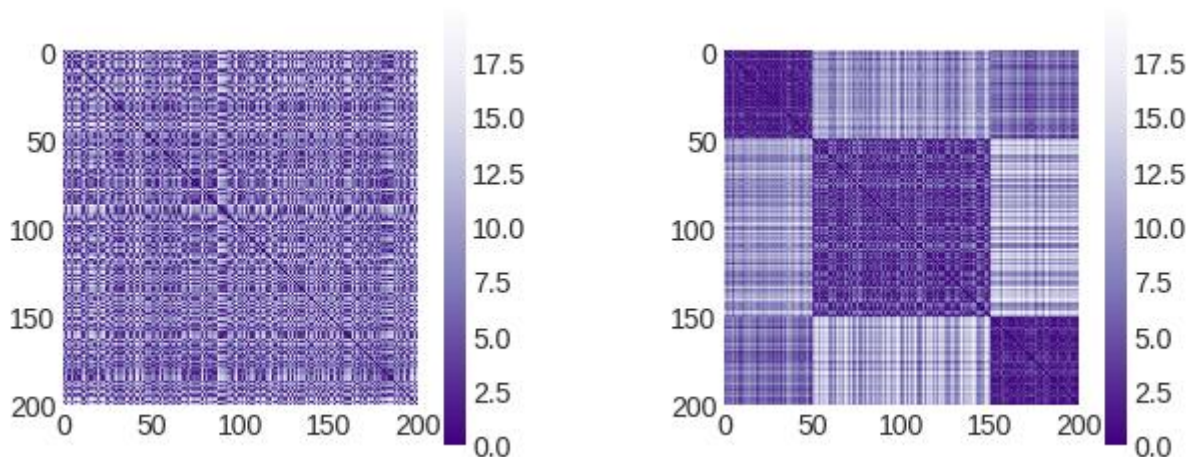


Рис. XX.33. Матрицы попарных расстояний объектов выборки: случайный порядок объектов (слева), после кластеризации (справа).

**2. Кластеризация для генерации признаков.** Как и многие другие методы обучения на неразмеченных данных (USL), кластеризацию можно применять для получения признаков в задачах с метками. Такие признаки хороши, например, в задачах с частичной разметкой (Semi-Supervised Learning). Сначала делается кластеризация, если кластеризация чёткая, то кластеры объектов выборки кодируются методом ONE: создаются бинарные характеристические признаки, их число совпадает с числом кластеров,  $i$ -й признак равен 1 тогда и только тогда, когда объект принадлежит  $i$ -му кластеру. Если нечёткая, то каждый признак описывает степень принадлежности к соответствующему кластеру. Признаки, порождённые кластеризацией добавляются к исходным.

Можно также добавлять признаки, описывающие положение в кластере (например, расстояние до центра), строить подобные признаки для других кластеризаций, делать кластеризацию на признаковом подпространстве, выбирать другое кодирование и т.п.

Многие методы полезны как служебные в задачах другого типа!

**3. Кластеризация для сжатия.** Рассмотрим цветное изображение (см. рис. XX.34, слева), каждый пиксель задан тремя компонентами: RGB<sup>1</sup>, т.е. является точкой в 3х-мерном пространстве. Таким образом, изображение задаёт множество точек в  $\mathbb{R}^3$ . Сделаем кластеризацию этого множества, на рис. XX.34 справа разными оттенками синего представлены  $k=16$  получившихся кластеров, если теперь для каждого кластера вычислить центр, то ему также соответствует некоторый цвет. На рис. XX.34 в центре показано изображение, в котором пиксели одного кластера отображаются цветом, который соответствует центру, т.е. используется всего  $k=16$  цветов. Заметим, что изначально для кодирования изображения размером  $m \times n$  требовалось  $3 \cdot 8 \cdot mn$  бит (если каждую компоненту кодировать числом от 0 до 255). Теперь при  $k=2^s$  достаточно для  $mn$  позиций хранить номер кластера от 1 до  $k$  (это  $s$  бит) и цвета каждого из кластеров, т.е. хранить всего  $3 \cdot 8 \cdot k + mns$  бит. На рис. XX.34 видно, что визуально изображение не стало заметно хуже, при этом удалось его существенно «сжать».

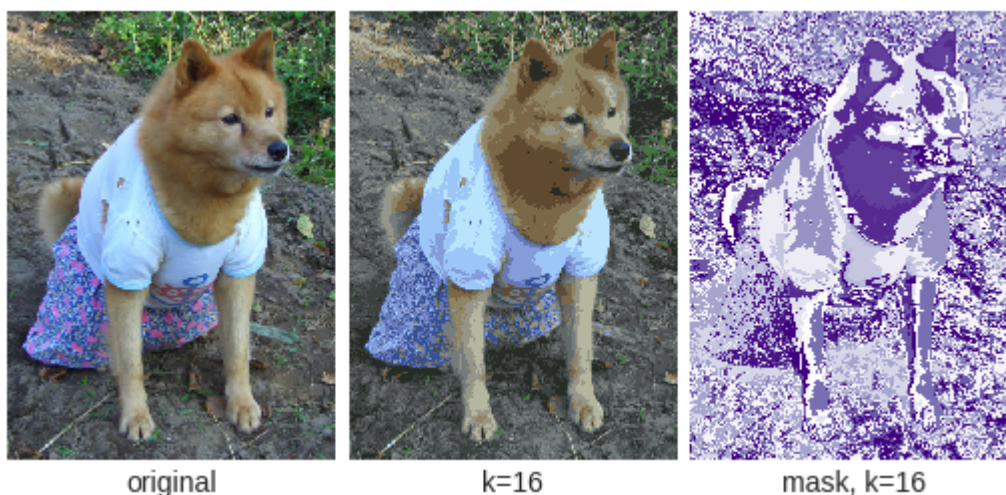


Рис. XX.34. Оригинальное изображение (слева), кластеризация пикселей (справа) и сжатое представление изображения (по центру).

Конечно, есть более экономные способы хранения изображений, но заметим, что мы просто использовали кластеризацию, причём к данным, которые могут

<sup>1</sup> <https://ru.wikipedia.org/wiki/RGB>



не иметь выраженной кластерной структуры. На рис. XX.35 показаны пиксели реального изображения в виде точек в RGB-пространстве, не видно ярко выраженных отдельно стоящих сгустков точек. Более того, кластеры, если есть, скорее вытянуты, а мы использовали для кластеризации алгоритм  $k$ -means, более подходящий для шаровидных кластеров, результат на рис. XX.36.

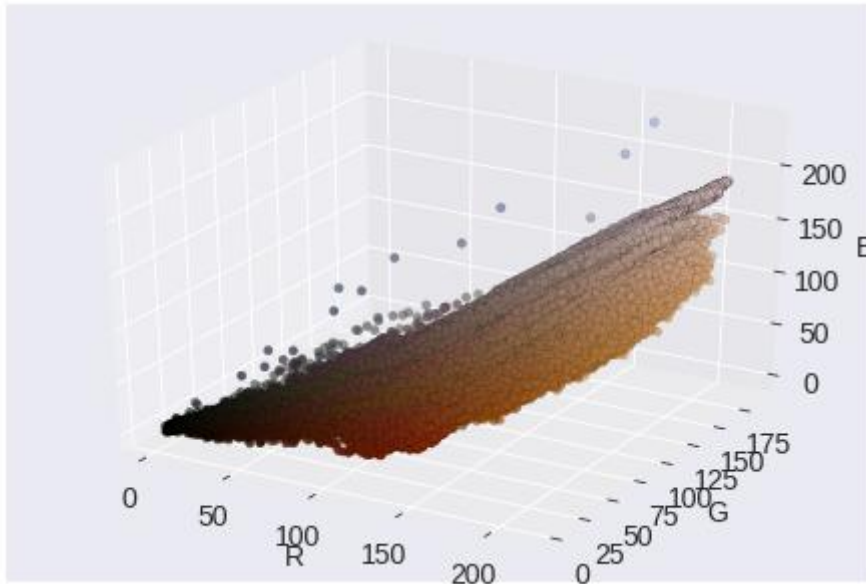


Рис. XX.35. Пиксели в RGB-пространстве.

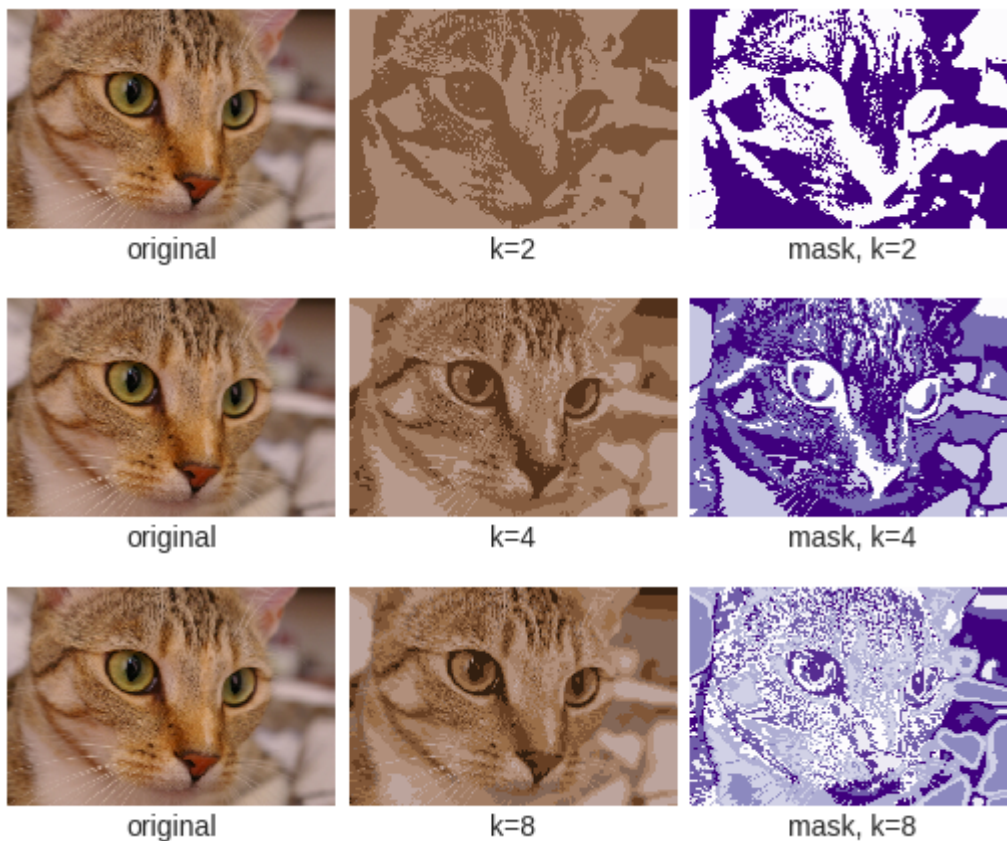


Рис. XX.36. Результаты сжатого представления при разном числе кластеров.



Результат сжатия, конечно, зависит от числа кластеров. На рис. XX.36 показаны результаты сжатия изображения котёнка при различном числе кластеров  $k \in \{2, 4, 8\}$ .

**4. Кластеризация для сегментации.** Покажем ещё одно применение матрицы связности. На рис. XX.34-36 была показана кластеризация пикселей в RGB-пространстве. Если использовать граф 4-соседства, в котором пиксель с координатами  $(i, j)$  соединяется рёбрами со своими соседями по сетке пикселей:  $(i \pm 1, j \pm 1)$  (соседей 4, если только пиксель не лежит на краю сетки), то в иерархической кластеризации мы получаем результат показанный на рис. XX.37. Здесь справа оттенками синего показаны полученные кластеры – связные множества пикселей, по центру они закрашены своими средними цветами. Результат похож на сегментацию – разделение изображения на составные примитивные части (майка, две лапы, голова и т.п.), неидеальность разделения связана с небольшим числом кластеров и простотой используемого подхода.

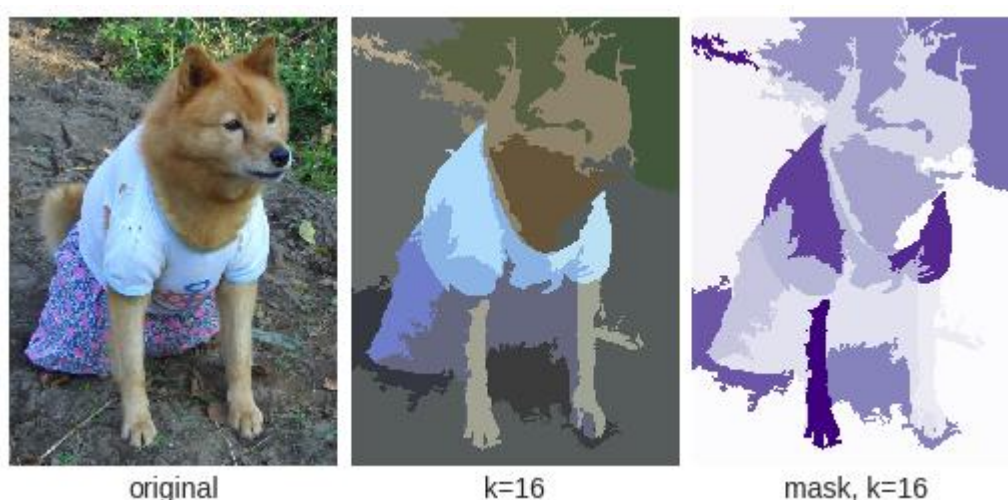


Рис. XX.37. Сегментации в изображениях по признакам (R, G, B) и графу 4-соседства. Исходное изображение (слева), результат кластеризации (справа), закрашивание каждого кластера средним цветом (по центру).

**5. Кластеризация для разделения объектов.** На рис. XX.38 показана спектральная кластеризация пикселей, каждый пиксель задаётся тремя компонентами (RGB), изображение довольно шумное, а граф-гиперпараметр метода определяется 4-соседством пикселей как в предыдущем примере (с соответствующим выбором весов). Предварительно сделана сегментация и на вход кластеризатору подаётся взвешенный граф, вершины которого описывают пиксели видимых соприкасающихся фигур. Видно, что кластеры визуально

соответствуют этим фигурам: на рис. XX.38 справа разные кластеры показаны разным цветом.



Рис. XX.38. Пример спектральной кластеризации пикселей: исходное изображение (слева) и кластеризация (справа).



Рис. XX.39. Другой пример спектральной кластеризации пикселей: исходное изображение (слева) и кластеризация (справа).

Однако, не стоит ждать от кластеризации чуда: здесь мы сообщили кластеризатору нужное число кластеров: 4. Кроме того, наши фигуры соприкасались незначительно. Если «наложить их друг на друга», как на рис. XX.39, то кластеризация уже не отделяет эти фигуры<sup>1</sup>. В любом случае, мы лишний раз продемонстрировали, что использование графа соседства может быть полезно, при кластеризации в исходном (RGB) пространстве без учёта соседства пикселей результата рис. XX.38 не получилось бы.

**6. Широкое применение ЕМ-алгоритма.** Мы рассмотрели ЕМ-алгоритм для разделения гауссовской смеси, но в машинном обучении он используется и в других задачах. Рассмотрим задачу оценивания вероятностей выпадения орла у нескольких монет<sup>2</sup>.

<sup>1</sup> Понятно, что кластеризатор не может определить их форму.

<sup>2</sup> Данная задача и её решение опубликовано в Nature. <https://www.nature.com/articles/nbt1406?pagewanted=all>

coin	tosses	coin	tosses	coin	tosses
0	0 [0, 0, 0, 0, 0, 0, 0, 0, 1, 0]	5	0 [0, 1, 0, 0, 0, 0, 1, 0, 0, 1]	10	1 [1, 1, 1, 1, 0, 0, 1, 0, 1, 0]
1	2 [0, 0, 1, 1, 1, 1, 1, 0, 1, 1]	6	0 [0, 0, 0, 0, 0, 0, 0, 0, 0, 1]	11	2 [1, 1, 1, 0, 1, 1, 0, 0, 1, 1]
2	0 [0, 0, 0, 0, 0, 0, 0, 0, 0, 1]	7	2 [0, 1, 1, 1, 1, 1, 1, 1, 1, 1]	12	2 [1, 1, 1, 1, 1, 0, 1, 1, 1, 1]
3	0 [0, 0, 0, 0, 1, 1, 0, 0, 0, 0]	8	1 [1, 1, 0, 0, 0, 1, 0, 1, 1, 1]	13	0 [0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
4	0 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]	9	0 [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]	14	1 [1, 0, 0, 1, 1, 1, 0, 0, 0, 0]

Рис. XX.40. Протокол экспериментов с монетами.

Пусть есть  $k$  нечестных монет, у каждой своя вероятность выпадения орла:  $p_t$ ,  $t \in \{1, 2, \dots, k\}$ , с ними продельвается  $m$  экспериментов. В каждом эксперименте выбирается одна из монет, случайно с равными вероятностями выбора  $p = 1/k$ . (можно предложить метод решения и для нечестного выбора). После этого выбранная монета подбрасывается  $n$  раз. Таким образом, в результате есть бинарная  $m \times n$ -матрица  $\|x_{ij}\|$  исходов:  $x_{ij} = 1$ , если в  $i$ -м эксперименте при  $j$ -м броске выпал орёл. Требуется оценить вероятности выпадения орлов  $p_t$ ,  $t \in \{1, 2, \dots, k\}$ . ЕМ-алгоритм для решения такой задачи запишется следующим образом:

Задаёмся случайными инициализациями:  $p_t \in (0, 1)$ ,  $t \in \{1, 2, \dots, k\}$ .

Е-шаг. Для каждого эксперимента оцениваем вероятность того, что он был получен с помощью соответствующей монеты:

$$r_{it} \sim \prod_{j=1}^n p_t^{x_{ij}} (1 - p_t)^{(1-x_{ij})}.$$

На основе этих значений можно оценить вероятность, что в эксперименте использована определённая монета:

$$\gamma_{it} = \frac{r_{it}}{\sum_s r_{is}}.$$

М-шаг. Пересчитываем вероятности выпадения орлов для монет:

$$p_t = \frac{\sum_{i=1}^m \left( \gamma_{it} \cdot \sum_{j=1}^n x_{ij} \right)}{\sum_{i=1}^m (\gamma_{it} \cdot n)}, \quad t \in \{1, 2, \dots, k\}.$$

На рис. XX.40 показаны результаты  $m=15$  экспериментов с  $k=3$  монетами, в каждом эксперименте было  $n=10$  бросков, в столбце «coin» показан номер выбранной в эксперименте монеты, а в столбце «tosses» – результаты её бросков: 1 – орёл, 0 – решка. При этом  $p_1=0.1$ ,  $p_2=0.5$ ,  $p_3=0.75$ . У нас не так много всего бросков ( $mn=150$ ), чтобы точно оценить вероятности, тем не менее, как видно на рис. XX.41, на 5й итерации получают довольно точные оценки (пунктиром показаны истинные значения вероятностей).

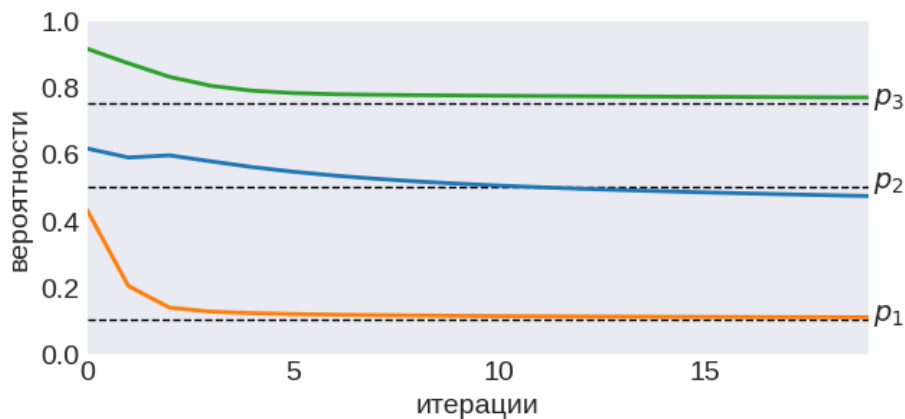


Рис. XX.41. Оценка вероятностей выпадения орла на итерациях EM-алгоритма.

**7. Кластеризация текстовых сообщений.** Приведём пример такой кластеризации, для иллюстрации использовали набор данных CLINC<sup>1</sup>, содержащий фразы из диалогов. В начале каждый текст был переведён в своё векторное представление (embedding) с помощью модели Sentence Bert<sup>2</sup>. Затем с помощью метода UMAP была понижена размерность до двух, на рис. XX.42 показаны проекции текстов в используемое двумерное пространство. Далее с помощью метода  $k$ -means была произведена кластеризация (при кластеризации реальных текстов в одном бизнес-проекте использовался DBSCAN, поскольку число кластеров сложно оценить экспертно, но в примере с датасетом CLINC известно число классов в нём). Результаты показаны на рис. XX.42, поскольку

<sup>1</sup> Отметим, что такая задача решалась в рамках проекта построения графа диалога в одном современном стартапе. Но здесь для иллюстрации выбран датасет с «достаточно чистыми данными» <https://paperswithcode.com/dataset/clinc150>

<sup>2</sup> Из библиотеки SentenceTransformer: model='sbnet', mode='sentence-t5-base'.

кластеров здесь много (более 100), цветными крупными точками показаны лишь первые 20, остальные – маленькими чёрными. На левой части рис. XX.42 показаны результаты кластеризации, на правой – результаты исходной разметки (также выбраны 20 классов, они не связаны с 20 кластерами на левой части). В данном случае, действительно, пространство объектов имеет кластерную структуру.

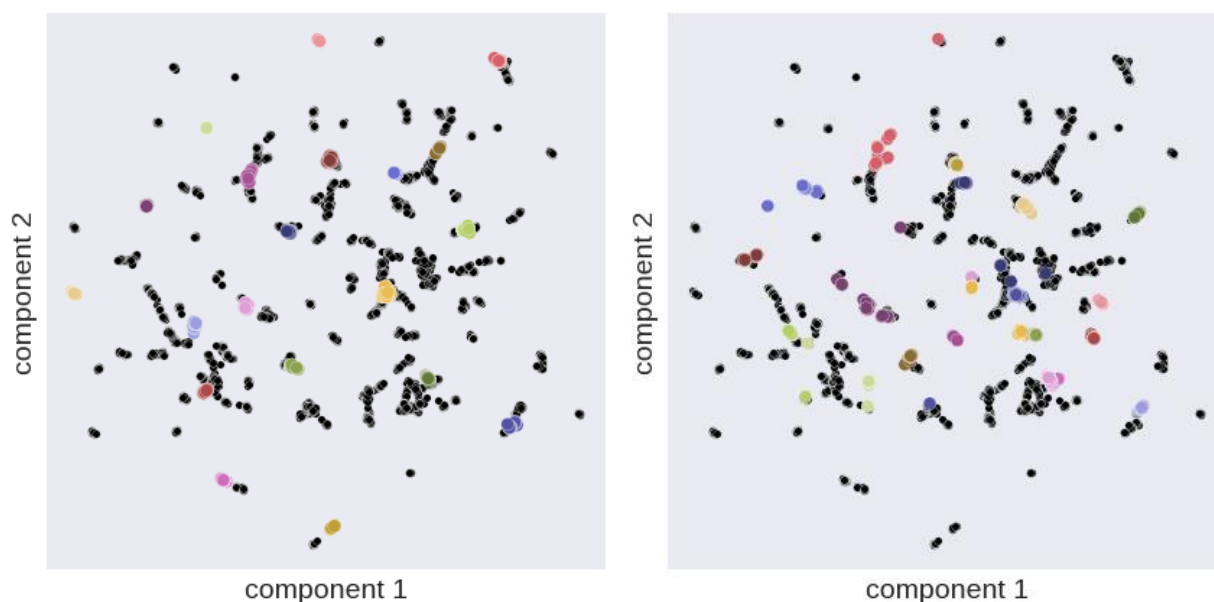


Рис. XX.42. Пространство кластеризации: слева – с метками полученных кластеров, справа – с метками исходных классов (цветом показаны 20 классов/кластеров, остальные – чёрные точки).

Приведём примеры полученных кластеров. Первый кластер содержит фразы

```
123, when should i change my car oil
123, do i need to change my oil
123, how often do you have to change your oil
123, how do i know when i have to change my oil
123, when do i need to change my oil
123, when am i due for an oil change
123, how long before i neet to get my oil changed
123, at how many miles am i required to get my oil changed
123, when should my oil get changed
....,
```

(в начале каждой указан номер класса в исходном датасете), что очень логично, все они являются вопросами о замене масла. Второй кластер состоит из фраз

```
61, if i were english how would i say subway
61, i would i say subway if i were english
61, in england how do they say subway
61, what do you call a subway if you were english
61, how would i say if i were english subway
```

```

51, use a different accent
51, try using a different accent
51, speak in a different accent
51, change the accent you're speaking in
... ,

```

здесь часть фраз о произнесении слова «метро» на английском языке, а часть о смене акцента, очевидно, что логичнее было бы разбить этот кластер на два подкластера (что соответствует и исходной классификации). Аналогичная ситуация и со следующим кластером

```

53, remind me to call my mother saturday morning
53, set a reminder to call my mom
53, remind me friday to call my mother
26, can you text elizabeth and tell them i forgot to bring drinks
26, text wenona and tell her we will be there tomorrow
26, text christopher and tell him i will stop by
26, text audrey and tell her i will be there soon
26, text christy and ask her what she wants for dinner
26, can you text christopher and tell them im on my way
... ,

```

но стоит признать, что кластеризация довольно логичная. Например, здесь в один кластер попали фразы с просьбами сообщить информацию: напомнить позвонить маме и написать что-то другому. На практике очень трудно, чтобы кластеризация текстов работала «как нужно» и тексты кластеров были объединены смыслом как это требует семантика решаемой задачи.

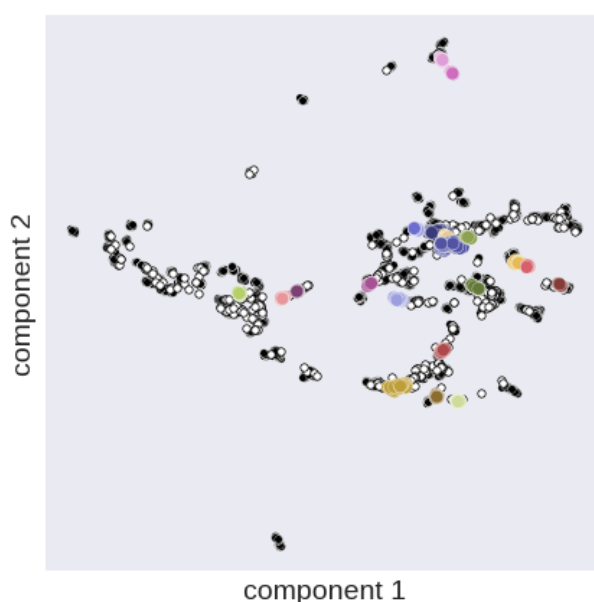


Рис. XX.43. Результат DBSCAN.

Интересно, что при использовании DBSCAN многие кластеры получаются такими же, например второй, что объясняется близостью представлений



входящих в них текстов (т.е. это проблема кодировщика, в данном случае Sentence Bert, а не кластеризатора). На рис. XX.43 белыми точками отмечен шум: кажется, что это основная масса точек, но, на самом деле, шумовых текстов менее 3%, просто все кластеры сильно концентрированные (а на рисунке кажется, что это одна цветная точка). Также отметим, что тут немного другое пространство для кластеризации: 4е компоненты UMAP вместо 2х (так получалось более высокое качество кластеризации) и на рис. XX.43 изображена проекция на первые две компоненты (что также вводит в заблуждение при анализе рисунка).

### Задачи и вопросы

1. Докажите, что  $\text{inertia}$  (XX.1) не возрастает на итерациях алгоритма k-средних<sup>1</sup>.
2. Эквивалентны ли метод Single Link и кластеризация на основе минимального остовного дерева MST (т.е. при заданном числе кластеров, всегда ли совпадают кластеризации, полученные этими методами)?
3. Обоснуйте формулу для вычисления расстояния Варда.
4. Выведите формулу для пересчёта матриц ковариаций в ЕМ-алгоритме для модели GMM (остальные формулы мы вывели).
5. Для скорректированных внешних показателей качества кластеризации выведите формулы для математических ожиданий.
6. В приложениях предложена идея визуализации матрицы попарных расстояний. Предложите способ упорядочивания объектов внутри кластеров.
7. Предложите метод решения задачи оценивания вероятностей выпадения орла у нескольких монет в случае, когда у каждой монеты своя вероятность выбора для получения серии бросков с ней.

---

<sup>1</sup> Доказательство этого и многих других фактов можно найти в книге Simovici D. A. CLUSTERING: Theoretical and Practical Aspects. – World Scientific, 2021.

## Кластеризация: итоги

В заключение перечислим свойства изученных алгоритмов кластеризации и покажем их работу на стандартных модельных датасетах. Отметим, что мы изучили следующие общие подходы к кластеризации:

- итерационное уточнение кластеров, например, положения их центров (как в  $k$ -means) или мод (как в mean-shift),
- оценка плотности / распределений (например, в EM-алгоритм полноценно оценивает распределение при некоторых предположениях),
- иерархические подходы (разбиением или слиянием кластеров),
- использование графов и теории графов (от пороговых графов до графов соседства).

Есть ещё отдельный нейростево́й подход к кластеризации, который мы здесь не затронули<sup>1</sup>. На рис. XX.44 показаны, как разные кластеризаторы работают на разных модельных задачах. Отметим, что все значения гиперпараметров кластеризаторов были выбраны по умолчанию, кроме гиперпараметра «число кластеров» — ему сообщалось «правильное число». В такой схеме сравнения больше всего «пострадал» метод DBSCAN, т.к. подсказок он не получил, а значения гиперпараметров не соответствовали геометрии точечных конфигураций, на которых он запускался<sup>2</sup>.

Метод	Гиперпараметры	Использование
k-means	Число кластеров	Простой и популярный. Много обобщений.  Когда кластеры равномо́щны и шарообразны.
Affinity propagation	Фактор забывания, вероятность стать экзemplяром	Не требует знания числа кластеров. Можно применять при большом числе. Большие требования к памяти и времени.

<sup>1</sup> Про кластеризацию можно также почитать в Дьяконов, А. Г. «Анализ данных, обучение по прецедентам, логические игры, системы WEKA, RapidMiner и MatLab (практикум на ЭВМ кафедры математических методов прогнозирования)» МАКСПресс 2010 // <http://www.machinelearning.ru/wiki/images/7/7e/Dj2010up.pdf>

Также есть серия постов про нестандартные методы кластеризации: <https://habr.com/ru/post/322034/>

<sup>2</sup> В аналогичном сравнении [https://scikit-learn.org/stable/auto\\_examples/cluster/plot\\_cluster\\_comparison.html](https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html) более «чистые» модельные задачи и по ним можно сделать немного другие выводы.

		По матрице расстояний, можно использовать произвольные расстояния.
Mean-shift	Параметр ядра	Нахождение мод плотности, детектирование выбросов. Может строить много кластеров. Годится для произвольных распределений.
Spectral clustering	Число кластеров, число компонент (новых признаков), параметр ядра	Основан на спектральной теории графов. Использует граф расстояний / матрицу сходства. Идеален для большинства модельных задач, но иногда преподносит сюрпризы.  Плохо масштабируется. Может занимать много времени.
Ward hierarchical / Agglomerative clustering	Число кластеров / тип слияния и метрика / матрица связности	Результат – дендрограмма, может быть много кластеров. Можно добавить ограничения на связи  Хорошо масштабируется
DBSCAN	Порог близости, число объектов в окрестности	Подходит для сложной кластерной геометрии. Детектирует выбросы. Не требует задания числа кластеров. Хорошо масштабируется.  Неудобные гиперпараметры, требуется однородная плотность объектов. Может быть не самая быстрая реализация.
Gaussian mixtures	Число компонент	Специальный случай восстановления смеси распределений.  Кластерная геометрия соответствует расстоянию Махаланобиса.

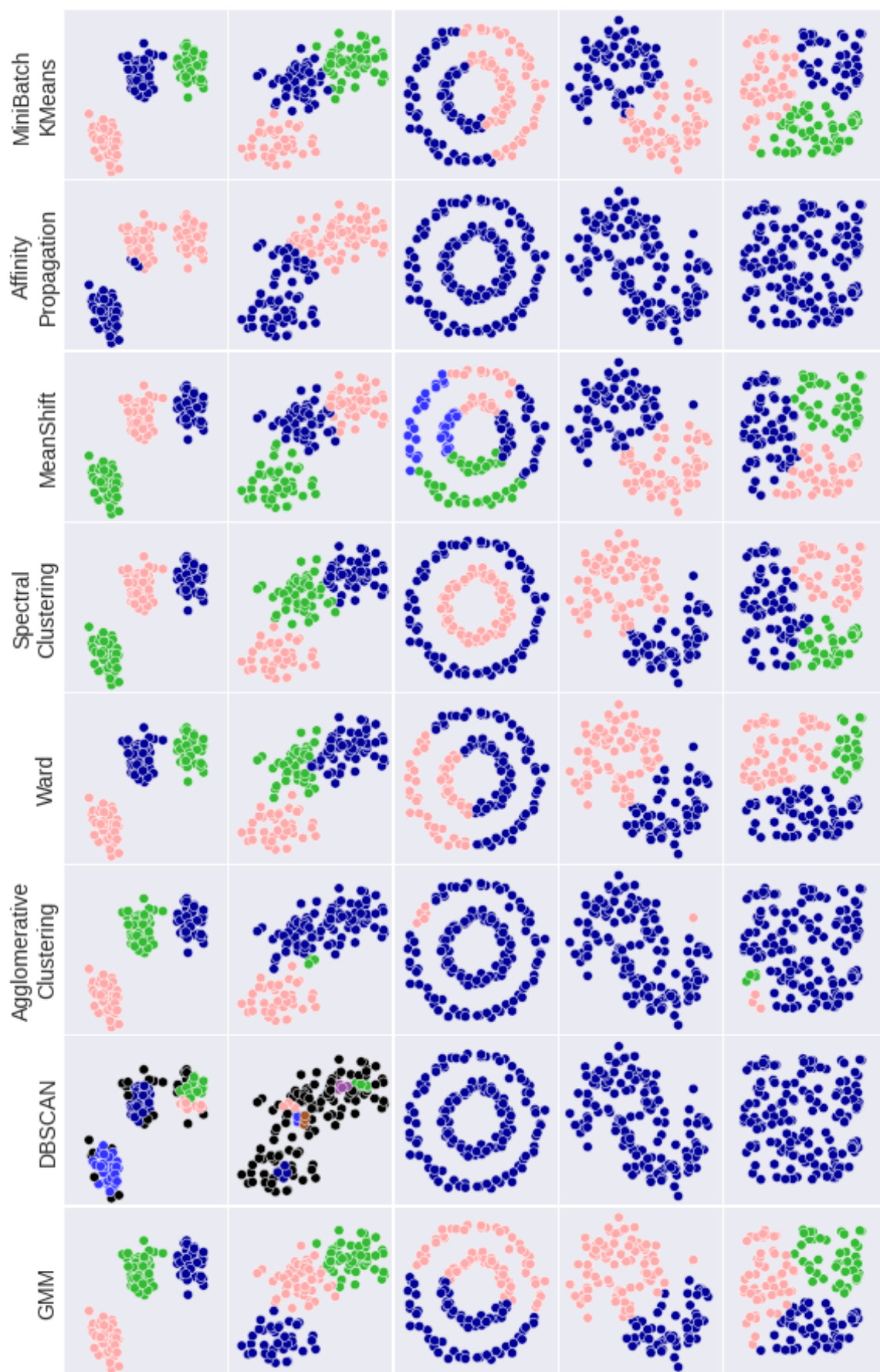


Рис. XX.44. Результаты кластеризации разными алгоритмами.

Спасибо за внимание к книге!  
Замечания по содержанию, замеченные ошибки  
и неточности можно написать в телеграм-чате  
<https://t.me/Dyakonovsbook>